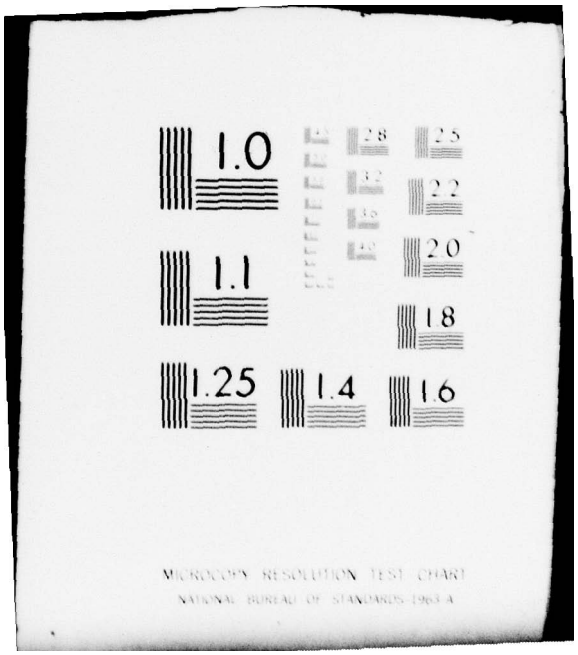
AD-A076 264 STANFORD UNIV CALIF DEPT OF COMPUTER SCIENCE F/G 12/1
UPPER AND LOWER BOUNDS ON TIME-SPACE TRADEOFFS IN A PEBBLE GAME--ETC(U)
JUL 79 T LENGAUER
UNCLASSIFIED STAN-CS-79-745 N00014-76-C-0688
NL

1 OF 1
AD A076264

END
DATE FILMED
11-79
DDC


1.0
1.1
1.25
1.4
1.6
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.8
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963-A

12

UPPER AND LOWER BOUNDS ON TIME-SPACE TRADEOFFS IN A PEBBLE GAME

by

Thomas Lengauer

STAN-CS-79-745
July 1979

DEPARTMENT OF COMPUTER SCIENCE
School of Humanities and Sciences
STANFORD UNIVERSITY

(9) Technical repts

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
(14) 1. REPORT NUMBER STAN-CS-79-745	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) (6) Upper and Lower Bounds on Time-Space Tradeoffs in a Pebble Game		5. TYPE OF REPORT & PERIOD COVERED technical, July 1979
		6. PERFORMING ORG. REPORT NUMBER STAN-CS-79-745
7. AUTHOR(s) (10) Thomas Lengauer		8. CONTRACT OR GRANT NUMBER(s) (15) N00014-76-C-0688
9. PERFORMING ORGANIZATION NAME AND ADDRESS Department of Computer Science Stanford University Stanford, California 94305 USA		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS (12) 83
11. CONTROLLING OFFICE NAME AND ADDRESS Office of Naval Research Department of the Navy Arlington, Va 22217		12. REPORT DATE (11) July 1979
		13. NUMBER OF PAGES 82
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) ONR Representative - Philip Surra Durand Aeromautics Building, Room 165 Stanford University Stanford, Ca. 94305		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Releasable without limitations on dissemination.

This document has been approved for public release and sale; its distribution is unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

pebbles, directed acyclic graphs, time-space tradeoffs, superconcentrators

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

(see reverse side)

094 120 JCB

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

We derive asymptotically tight time-space tradeoffs for pebbling three different classes of directed acyclic graphs. Let N be the size of the graph, S the number of available pebbles, and T the time necessary for pebbling the graph.

(a) A time-space tradeoff of the form

$$ST = \theta(N^2)$$

is proved for pebbling (using only black pebbles) a special class of permutation graphs that implement the bit reversal permutation. If we are allowed to use black and white pebbles the time-space tradeoff is shown to be of the form

$$T = \theta(\frac{N^2}{S^2}) + \theta(N).$$

(b) A time-space tradeoff of the form

$$T = S\ \theta(\frac{N}{S})^{\theta(\frac{N}{S})}$$

is proved for pebbling a class of graphs constructed by stacking superconcentrators in series. This time-space tradeoff holds whether we use only black or black and white pebbles.

(c) A time-space tradeoff of the form

$$T = S\ 2^{2^{\theta(\frac{N}{S})}}$$

is proved for pebbling general directed acyclic graphs with only black or black and white pebbles.

UPPER AND LOWER BOUNDS ON TIME-SPACE TRADEOFFS IN A PEBBLE GAME

Thomas Lengauer

Department of Computer Science
Stanford University
Stanford, California 94305

July 1979

KEYWORDS: pebbles, directed acyclic graphs, time-space, superconcentrators.

This work was supported by the German Academic Exchange Service (DAAD). Printing and distribution was supported by the Office of Naval Research under contract N00014-76-C-0688.

Abstract

We derive asymptotically tight time-space tradeoffs for pebbling three different classes of directed acyclic graphs. Let N be the size of the graph, S the number of available pebbles, and T the time necessary for pebbling the graph.

(a) A time-space tradeoff of the form

$$ST = \Theta(N^2)$$

is proved for pebbling (using only black pebbles) a special class of permutation graphs that implement the bit reversal permutation. If we are allowed to use black and white pebbles the time-space tradeoff is shown to be of the form

$$T = \Theta(\frac{N^2}{S^2}) + \Theta(N).$$

(b) A time-space tradeoff of the form

$$T = S\ \Theta(\frac{N}{S})^{\Theta(\frac{N}{S})}$$

is proved for pebbling a class of graphs constructed by stacking superconcentrators in series. This time-space tradeoff holds whether we use only black or black and white pebbles.

(c) A time-space tradeoff of the form

$$T = S\ 2^{2^{\Theta(\frac{N}{S})}}$$

is proved for pebbling general directed acyclic graphs with only black or black and white pebbles.

To my parents and my brother

ACKNOWLEDGMENTS

I am deeply indebted to my advisor, Bob Tarjan. Through his illuminating and motivating lectures he first interested me in complexity theory. Through a very rewarding master's project he introduced me to applied research in the area. As a teaching assistant for his courses I got the chance to give lectures and to try to follow his uncompromising standards of teaching. And finally, after the suggestion of an excellent thesis topic, he enabled me through a period of exciting joint research to put together this dissertation. Except for Section 2.1–2.3, Chapter 3 and Section 4.3 all of the results in this document have been established in joint research with Bob.

I would like to express my thanks to Don Knuth and Nick Pippenger. Both read this thesis with interest and scrutiny and provided many useful suggestions. Nick Pippenger had a particular influence on Chapter 2. Firstly he suggested the investigation of bit reversal graphs, and secondly he contributed simplifications in the proof of Theorem 2.3.2 that greatly reduced the size of Section 2.3 and improved the constant factors involved. Furthermore he made me aware of a convexity argument that avoids the use of multidimensional calculus in the proof of Theorem 4.2.6.

Some of the proofs have been verified using the formula manipulator of the MACSYMA-Consortium.

The financial support of my study through the German Academic Exchange Service and the German National Fellowship Foundation is gratefully acknowledged.

Of all those who directly or indirectly helped me towards my degree my parents certainly made the greatest sacrifice. With unsurpassable love, and quite unselfishly disregarding their own longings, they supported my program from the beginning with mind and heart. Across half the globe my family stayed and is continuing to stay close to me in spirit; their love is an inexhaustible source of strength and comfort.

Finally, I will always be grateful to Becky & Mike, Bill, Elena & Juan, Greg, and Rosa & Jorge, who all showed me how beautiful international friendship can be. They made my four years at Stanford an unforgettable period in my life, and they are the main reason why it is so hard to leave here.

TABLE OF CONTENTS:

LIST OF FIGURES:

NOTATIONAL DEFINITIONS:

$f(n) = O(g(n))$	iff there are constants $n_0 > 0$ and $c > 0$ such that for all $n > n_0$, $f(n) \leq c\,g(n)$.
$f(n) = \Omega(g(n))$	iff there are constants $n_0 > 0$ and $c > 0$ such that for all $n > n_0$, $f(n) \geq c\,g(n)$.
$f(n) = \Theta(g(n))$	iff $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$.
$f(n) = o(g(n))$	iff $\lim_{n\to\infty} f(n)/g(n) = 0$.
$f(n) = \omega(g(n))$	iff $\lim_{n\to\infty} g(n)/f(n) = 0$.
$\log x$	Binary logarithm of x.
$\ln x$	Natural logarithm of x.
$\exp(x)$	Same as e^x.
$\lfloor x \rfloor$	The greatest integer y satisfying $y \leq x$.
$\lceil x \rceil$	The smallest integer y satisfying $y \geq x$.
$[i, j]$	The interval of integers $\{i, i+1, \ldots, j-1, j\}$. If $j < i$ then $[i, j]$ is the empty set.

As if we could kill time without injuring eternity!
—HENRY DAVID THOREAU (1817-1862)

1 INTRODUCTION

We study time-space tradeoffs in a pebble game defined in [HP70] and [Co73]. The game is played on directed acyclic graphs with bounded in-degree according to the following rules:

(i) A pebble may be removed from a vertex at any time.

(ii) If all predecessors of an unpebbled vertex v are pebbled, a pebble may be placed on v.

(iii) If all predecessors of an unpebbled vertex v are pebbled, a pebble may be moved from a predecessor of v to v.

The object of the game is to pebble each vertex in the graph at least once. (For a comparison of slightly differing definitions of the pebble game see [EL78].)

The pebble game has been used to model register allocation ([Se75]), to study flowcharts and recursive schemata ([HP70]), and to analyze the relative power of time and space as Turing-machine resources ([Co73], [HPV77]). Furthermore it has been used to derive time-space tradeoffs for several important algorithmic concepts such as linear recursion ([Cha73], [SS77]), Fast Fourier-Transform ([SS78a], [To78]), matrix multiplication ([To78]), and integer multiplication ([SS78b]). We are interested in the relative power of time and space as resources in the pebble game.

The *space* S required by the pebbling is the maximum number of pebbles ever on the graph simultaneously; the *time* T required is the number of applications of rules (ii) and (iii), i.e., the number of pebble placements. (One could also count both placements and removals. The results would then change by at most a factor of 2.) The *size* N of the graph is the number of its vertices.

In [HPV77] it is shown that any graph of size N can be pebbled using only $O(N/\log N)$ pebbles and in [PTC77] a proof is given that for certain graph families $\Omega(N/\log N)$ pebbles are necessary to pebble all vertices.

Recently interest arose in the study of precise time-space tradeoffs in the pebble game. The rationale is that space savings are only feasible if the corresponding sacrifice in computing time is reasonable. It is of interest to know how much the time increases as the amount of available space, i.e., pebbles, is decreased.

The results given in [HPV77] and [PTC77] limit the range of interest for S to $\Omega(N/\log N) = S \leq N$. Two trivial observations about the pebbling time impose a corresponding limitation on the range of interest for T.

(1) Any graph of size N can be pebbled with N pebbles in time N (in topological order).

(2) If a graph G of size N can be pebbled with S pebbles at all then it can be pebbled with S pebbles in time $T \leq \sum_{0 \leq k \leq S} \binom{N}{k} \leq 2^N$.

(For the proof of (2) observe that the sum in (2) equals the number of different configurations of at most S pebbles on G and therefore any strategy for pebbling G whose length is greater has to repeat some configuration and can thus be shortened.) The range of interest for T is thus $2^N \geq T \geq N$.

Clearly T has to increase if S is decreased. The main open question in this area is how much T increases as S is decreased from N towards $\Omega(N/\log N)$. It is reasonable to make the following conjecture.

Conjecture C: There are graphs of size N that can be pebbled with $O(N/\log N)$ pebbles only in a time that grows superpolynomially in N.

Theoretical computer science often assumes that polynomial time algorithms are feasible whereas algorithms whose time complexity exceeds any polynomial are not. In this sense the conjecture asserts that there are graphs for which space savings of $S = O(N/\log N)$, though possible, are infeasible.

If Conjecture C is true it makes sense to look for a function $S_J(N)$ such that $\Omega(N/\log N) \leq S_J(N) \leq N$ and $S_J(N)$ has the following two properties:

(a) If $S \geq c_1 S_J(N)$ then each graph of size N can be pebbled with S pebbles in time T where T grows only polynomially in N.

(b) If $S \leq c_2 S_J(N)$ then there are graphs of size N that can only be pebbled with S pebbles in superpolynomial time.

(Here $c_1, c_2 > 0$ are suitable constants.) The threshold function $S_J(N)$ locates the asymptotic range for S where the "jump" from polynomial to superpolynomial time occurs in the pebble game. Space savings of $S \geq c_1 S_J(N)$ are always feasible, whereas there are graphs for which space savings of $S \leq c_2 S_J(N)$ are infeasible. (Note that because of the asymptotic nature of the analysis, $S_J(N)$ is not uniquely defined. With any function $S_J(N)$ that fulfills properties (a) and (b) above any function that is $\Theta(S_J(N))$ also fulfills the properties. Indeed, the location of the "jump" from polynomial time to superpolynomial time can only be defined asymptotically.)

Not much progress has been made heretofore towards proving Conjecture C. [Cha73], [Rei78], [SS77], [SS78a], [SS78b] and [To78] discuss time-space tradeoffs for natural and constructible graph families. However, all these families can be pebbled with $S = \Theta(N/\log N)$ pebbles in linear time.

[EL78], [Li78] and [PT77] discuss graph families whose pebbling time increases explosively from linear to superpolynomial at a certain point, as the number of available pebbles is decreased. However, this point lies in the range $S = o(N/\log N)$ and for $S = \Omega(N/\log N)$ these graphs can again be pebbled in linear time.

In [Pip78] Pippenger proves the most dramatic time-space tradeoff known heretofore for a family of graphs that is straightforward and can easily be constructed. It has the form

$$T = \frac{N}{2} \log \frac{N}{2S} + O(N).$$

For $S = O(N/\log N)$ we have $T = \Omega(N \log\log N)$.

[To78] and [Rei78] discuss families of graphs that are much harder to construct but have more dramatic time-space tradeoffs for $S = \Omega(N/\log N)$. However, none of the lower bounds they prove reach

$$T = \Omega(N^{1+\epsilon}), \qquad \text{for any } \epsilon > 0$$

if $S = \Theta(N/\log N)$. They are thus far from exhibiting a superpolynomial blow-up in time in the relevant range of S.

Independently from the work presented in this thesis Reischuk proves in [Rei78] an upper bound on S_J. He shows that any graph of size N can be pebbled with $S = \Omega(N/\log^r N)$ pebbles ($r \in \mathbb{N}$) in time T where

$$T = N\ 2^{2^{O(\frac{N}{S}\log\frac{N}{S})}}. \tag{1}$$

Therefore we cannot expect to find a graph of size N whose pebbling with $S \geq cN/\log\log\log N$ pebbles takes superpolynomial time ($c > 0$ is any positive constant). Thus

$$S_J = o(N/\log\log\log N).$$

(It is possible that the tighter upper bound

$$S_J = O(N\log\log\log N/\log\log N)$$

is implied by Reischuk's algorithm. As he states his result, this bound which would follow directly from an analysis of equation (1) cannot be inferred, however, because equation (1) is only proved for the values $S = \log^r N$ for $r \in \mathbb{N}$.)

This thesis proves Conjecture C, locates S_J, and contributes the analysis of a new straightforward graph family with a rather dramatic time-space tradeoff.

Chapter 2 discusses an easily constructible graph family whose time-space tradeoff is more dramatic than that discussed in [Pip78]. The tradeoff has the form

$$ST = \Theta(N^2).$$

For $S = O(N/\log N)$ we have $T = \Omega(N\log N)$.

Chapter 3 proves Conjecture C by exhibiting a two-parameter graph family with the time-space tradeoff

$$T = S\ \Theta(\frac{N}{S})^{\Theta(\frac{N}{S})}.$$

T is thus superpolynomial in N for $S \leq cN\log\log N/\log N$ ($c > 0$ is a suitably small constant). For $S = O(N/\log N)$ we have $T = N^{\Omega(\log\log N)}$.

Chapter 4 locates S_J by proving a time-space tradeoff for pebbling general graphs of the form

$$T = S\ 2^{2^{\Theta(\frac{N}{S})}}.$$

Thus $S_J = \Theta(N/\log\log N)$. For $S = \Theta(N/\log N)$ we have $T = 2^{N^{\Omega(1)}}$, i.e., T is exponential.

There is a variation of the pebble game which has been studied in [CS76], [GT78] and [Me78]. In this variation pebbles of two colors, black and white, are available. Three additional rules govern the use of white pebbles:

(iv) A white pebble can be placed on an empty vertex at any time.

(v) A white pebble can be removed from a vertex v if all its predecessors are pebbled.

(vi) If all but one of the predecessors of a vertex v having a white pebble are pebbled, then the white pebble can be moved from v to its unpebbled predecessor.

The object of the game is to finish with no pebbles on the graph starting with no pebbles on the graph and pebbling each vertex at least once. The space S required by the pebbling is the maximum number of pebbles ever on the graph simultaneously and the time T required is the number of applications of rules (ii), (iii), (iv) and (vi), i.e., the number of pebble placements. Again the size N of the graph is the number of its vertices. We call this game the *black & white* pebble game. The white pebbles represent non-deterministic guesses during a computation. They are easy to place, since guesses can easily be made, but hard to remove, since guesses have to be checked. In this sense the black & white pebble game is the non-deterministic version of the usual (black) pebble game.

[CS76] introduces the black & white pebble game and proves a lower bound on the number of black and white pebbles necessary to pebble a certain graph family called *pyramid graphs*. [Me78] extends this lower bound to all directed acyclic graphs: If a graph can be pebbled with k black and white pebbles then it can be pebbled with $O(k^2)$ black pebbles. [GT78] shows that the $\Omega(N/\log N)$ space lower bound from [PTC77] carries over to the black & white pebble game (with different constant factors). The main open questions in this area are the following.

(1) Are there graphs for which black and white pebbles save more than a constant factor of space over black pebbles?

(2) If so, what is the maximum savings possible? ([Me78] shows that it has to be of the order of the square root.)

In addition to these questions the study of time-space tradeoffs in the black & white pebble game is of interest.

Although we are primarily interested in the black pebble game, it turns out that for the three graph families we consider in this thesis the time-space tradeoffs in the black & white pebble game are straightforward extensions of the time-space tradeoffs in the black pebble game. We therefore include also proofs of the following results.

In Chapter 2 a time-space tradeoff of the form

$$T = \Theta(\frac{N^2}{S^2}) + \Theta(N).$$

is derived for pebbling bit reversal graphs with black and white pebbles.

In Chapter 3 it is shown that the time-space tradeoff for pebbling stacks of superconcentrators with black and white pebbles is asymptotically identical to the one for using only black pebbles, i.e., it has the form

$$T = S\ \Theta(\frac{N}{S})^{\Theta(\frac{N}{S})}.$$

In Chapter 4 it is shown that the time-space tradeoff for pebbling general directed acyclic graphs with black and white pebbles is asymptotically identical to the one for using only black pebbles, i.e., has the form

$$T = S\ 2^{2^{\Theta(\frac{N}{S})}}.$$

A summary of the research results presented in this thesis has been published in [LT79].

2 PERMUTATION GRAPHS

2.1 Introduction

Below we will define a particularly simple family of graphs, which are induced by permutations on N elements.

Definition 2.1.1: Let π be a permutation on N elements. The *permutation graph* $G(\pi)$ on N elements is the graph $G = (V, E)$ such that

$$V = \{\sigma_1, \sigma_2, \ldots, \sigma_N, \tau_1, \tau_2, \ldots, \tau_N\} \quad \text{and}$$
$$E = \{(\sigma_i, \sigma_{i+1}), (\tau_i, \tau_{i+1}) \mid 1 \leq i \leq N-1\} \cup \{(\sigma_i, \tau_{\pi(i)}) \mid 1 \leq i \leq N\}$$

For $1 \leq i \leq N$, σ_i is called the *i-th input vertex*, τ_i is called the *i-th output vertex*; σ_1 is called the *source*, τ_N is called the *sink*. The path consisting of the edges $(\sigma_1, \sigma_2), (\sigma_2, \sigma_3), \ldots, (\sigma_{N-1}, \sigma_N)$ is called the *input path*. The path consisting of the edges $(\tau_1, \tau_2), (\tau_2, \tau_3), \ldots, (\tau_{N-1}, \tau_N)$ is called the *output path*. (Figure 1 shows a typical permutation graph.)

A permutation graph on N elements thus has $2N$ vertices and a maximum in-degree of 2. One class of permutation graphs has already been studied in [HP70], [Cha73] and [SS77] and independently by the author. Sometimes called *ladder graphs*, they correspond to the permutation π defined by $\pi(k) = N + 1 - k$ and represent the memory allocation scheme in linear recursive programs. One can pebble ladder graphs fairly quickly: We have $T = \omega(N)$ only if $S = N^{o(1)}$.

2.2 The Upper Bound in the Black Pebble Game

It is straightforward to derive an upper bound on the time-space tradeoff for pebbling permutation graphs.

Fact 2.2.1: Each permutation graph can be pebbled with two pebbles.

Indeed, this fact is a special case of the following more general result.

Lemma 2.2.2: For each permutation graph on N elements an upper bound on the time-space tradeoff in the black pebble game is given by

$$T \leq \frac{N^2}{S-1} + N.$$

Proof: Assume that S pebbles are available. Reserve one pebble for the output path. In at most N steps pebble the vertices $\sigma_\pi{}^{-1}(1), \ldots, \sigma_\pi{}^{-1}(S-1)$ on the input path. Then in $S-1$ steps pebble τ_{S-1} with the pebble reserved for the output path. Move the pebbles on the input path in $N-S+1$ more steps as necessary to pebble the vertices $\sigma_\pi{}^{-1}(S), \ldots, \sigma_\pi{}^{-1}(2S-2)$ on the input path.

(This can be done as follows. The $S-1$ pebbles on the input path divide it into S intervals of vertices. All but the first interval are such that the first vertex in the interval is pebbled and all other vertices in the interval are not pebbled. The first (perhaps empty) interval is entirely free of pebbles. Let C be the set of the $S-1$ vertices on the input path to be pebbled next. We will pebble C in the order from small to large vertices. For each vertex $v \in C$ that is in the first interval there is another interval that does not contain any vertex in C. The pebble at the low end of this interval can be used to pebble v. When there is no vertex left to be pebbled in the first interval we can use the pebble on the next higher interval to pebble the first vertex in C that is in this interval. If there is more than one vertex to be pebbled in this interval then we can again use pebbles from other intervals that do not contain elements of C. Continuing in this fashion we can pebble C by placing pebbles on successively larger vertices on the input path. In this process only vertices that started out unpebbled are pebbled and each vertex is pebbled at at most once. Since $S-1$ vertices start out pebbled at most $N-S+1$ placements are made.)

Use $S-1$ more steps to pebble the vertex τ_{2S-2} on the output path. Continue in an analogous fashion.

This strategy pebbles t_N in $\lceil \frac{N}{S-1} \rceil$ phases, where each phase uses at most $N-S+1$ placements of pebbles on the input path except the first phase which may use N placements. Furthermore exactly N placements are made on the output path. Thus

$$T \leq (N-S+1)\left(\left\lceil \frac{N}{S-1} \right\rceil - 1\right) + 2N \leq \frac{N^2}{S-1} + N. \quad \blacksquare$$

Lemma 2.2.2 shows that for permutation graphs

$$ST \leq 3N^2.$$

For ladder graphs this upper bound is not at all tight. The question arises whether there is a family of permutation graphs for which this bound is tight up to a constant factor. Such a family would be interesting because it would in some sense represent the permutations that are most difficult to realize in serial computation schemes with restricted storage capacity. Section 2.3 shows that the bit reversal permutation is such a permutation.

2.3 The Bit Reversal Permutation

Let $0 \leq N = 2^n$ and for convenience let the set to be permuted be the set $I = \{i \mid 0 \leq i < N\}$. Let b be the bijective mapping $b : I \rightarrow \{0,1\}^n$ where $b(j)$ is the binary string of length n representing the number j.

Definition 2.3.1: Let $b(j) = b_{n-1} \ldots b_0$. The *bit reversal* of j (denoted by $\mathrm{rev}(j)$) is defined to be the number j' such that $b(j') = b_0 \ldots b_{n-1}$. (Figure 2 shows the bit reversal graph on $N = 32$ elements.)

The bit reversal permutation has the characteristic property that it scatters adjacent numbers approximately evenly over the interval I. This property is the key to the following lower bound proof.

Theorem 2.3.2: If $S \geq 2$ then pebbling the bit reversal graph on N elements with S pebbles takes at least time

$$T > \frac{N^2}{16S}.$$

Proof: The proof is trivial for $S > N/4$. Thus assume that $S \leq N/4$.

Choose the integer s such that

$$2S \leq 2^s < 4S.$$

Consider the output path divided into 2^{n-s} intervals of length 2^s. The j-th interval I_j $(0 \leq j < 2^{n-s})$ consists of the vertices $\tau_{j2^s}, \ldots, \tau_{(j+1)2^s-1}$.

Let t_j be the first time a pebble is placed on $\tau_{(j+1)2^s-1}$, i.e., on the highest vertex in I_j. Let $t_{-1} := 0$. Then $t_j > t_{j-1}$ for $0 \leq j < 2^{n-s}$. In order to find a lower bound on $t_j - t_{j-1}$ we observe that at time t_{j-1} the interval I_j is pebble-free and thus all 2^s vertices in I_j have to be pebbled between t_{j-1} and t_j. By definition of the bit reversal permutation the immediate predecessors of the vertices in I_j on the input path divide the input path naturally into $2^s - 1$ intervals of length 2^{n-s}. (The each immediate predecessor of a vertex in I_j defines the high limit of an interval. The intervals at the corners of the input path are disregarded.) At time t_{j-1} at most $S-1$ pebbles are on the input path. Thus at least $2^s - 1 - (S-1) \geq S$ intervals are pebble-free at t_{j-1}. All of them have to be pebbled completely before t_j. This takes at least $S \cdot 2^{n-s} > N/4$ placements. Therefore $t_j - t_{j-1} > N/4$ for $0 \leq j < 2^{n-s}$, and thus before time t_{2^n-1} at least $2^{n-s}N/4 > N^2/16S$ placements have to occur. ∎

2.4 Pebbling the Bit Reversal Graph With Black and White Pebbles

If we are allowed to use black and white pebbles to pebble the bit reversal graph then the vertices on the output path do not have to be pebbled in sequence. Rather we can place a certain number of white pebbles on the output path at the beginning and then pebble the intervals thus created on the output path independently of each other. If we use this idea we can exploit a regularity of the bit reversal permutation to speed up the pebbling such that

$$T = \Theta(\frac{N^2}{S^2}) + \Theta(N).$$

Theorem 2.4.1: The bit reversal graph on $N = 2^n$ elements can be pebbled with S pebbles ($3 \leq S \leq 3\sqrt{N}$) in time

$$T \leq 36\frac{N^2}{S^2} + 3N.$$

Proof: Let k be such that $3 \cdot 2^k \leq S < 3 \cdot 2^{k+1}$.

Put 2^k white pebbles on the outputs $\tau_0, \tau_{2^{n-k}}, \tau_{2 \cdot 2^{n-k}}, \tau_{3 \cdot 2^{n-k}}, \ldots, \tau_{(2^k-1) \cdot 2^{n-k}}$. This partitions the output path into 2^k intervals of length 2^{n-k}. The j-th interval is the interval $[\tau_{j2^{n-k}}, \tau_{(j+1)2^{n-k}-1}]$ for $0 \leq j < 2^k$. Each of these intervals we consider to be broken up into 2^{n-2k} chunks of length 2^k. The i-th chunk of the j-th interval consists of the vertices $[\tau_{j2^{n-k}+i2^k}, \tau_{j2^{n-k}+(i+1)2^k-1}]$ for $0 \leq i < 2^{n-2k}$.

We now reserve 2^k more pebbles (this time black ones) for the output path, one for each of the intervals created by the white pebbles. We will pebble the intervals created by the white pebbles in 2^{n-2k} phases. The i-th phase pebbles the i-th chunks of all intervals. We use 2^k more black pebbles on the input path to be able to pebble each chunk in one sweep.

Formally we assume inductively on i that the output vertices just before the beginning of the i-th chunks in all intervals, i.e., the vertices $\tau_{j2^{n-k}+i2^k-1}$ for all j such that $0 \leq j < 2^k$, have black pebbles on them. (In the initial case $i = 0$ the first vertex on each 0-th chunk has a white pebble on it and the argument proceeds in the same fashion.) We rearrange the 2^k pebbles on the input path in $N - 2^k$ steps (N steps in the 0-th phase) such that they are on the vertices

$$\sigma_{\mathrm{rev}(\mathrm{rev}(0)2^{n-k}+i2^k)}, \sigma_{\mathrm{rev}(\mathrm{rev}(0)2^{n-k}+i2^k+1)}, \ldots, \sigma_{\mathrm{rev}(\mathrm{rev}(0)2^{n-k}+(i+1)2^k-1)}$$

This enables us to sweep the black pebble on the $rev(0)$-th interval across the i-th chunk. Then by advancing each pebble on the input path one vertex we pebble the vertices

$$\sigma_{\mathrm{rev}(\mathrm{rev}(1)2^{n-k}+i2^k)}, \sigma_{\mathrm{rev}(\mathrm{rev}(1)2^{n-k}+i2^k+1)}, \ldots, \sigma_{\mathrm{rev}(\mathrm{rev}(1)2^{n-k}+(i+1)2^k-1)}$$

and can now sweep the black pebble on the $rev(1)$-th interval across the i-th chunk. Advancing pebbles on the input path in a suitable manner allows us to pebble the i-th chunks of all j intervals in the order $j = \mathrm{rev}(0), \mathrm{rev}(1), \mathrm{rev}(2), \ldots, \mathrm{rev}(2^k-1)$. It is easy to see that pebbling the i-th chunks of all j intervals takes

$$2^n - 2^k + (2^k - 1)2^k$$

placements on the input path if $i \neq 0$ and

$$2^n + (2^k - 1)2^k$$

placements on the input path if $i = 0$.

After all chunks are pebbled in this way the pebbles end up in a configuration that allows the white pebbles to be taken off the graph. The whole pebbling takes time

$$2^n + (2^n + (2^k - 1)2^k)2^{n-2k} - 2^k(2^{n-2k} - 1) \leq 36\frac{N^2}{S^2} + 3N. \qquad \blacksquare$$

Corollary 2.4.2: The bit reversal graph on N elements can be pebbled with S black and white pebbles ($2 \leq S \leq N+1$) in time

$$T \leq 36\frac{N^2}{S^2} + 5N.$$

Proof: For $S = 2$ see Lemma 2.2.2. If $3 \leq S \leq 3\sqrt{N}$ Theorem 2.4.1 applies. If $S > 3\sqrt{N}$ the strategy given in Theorem 2.4.1 takes time at most $5N$. ∎

The upper bound given in Corollary 2.4.2 can be matched asymptotically with a lower bound whose proof uses the characteristic property of the bit reversal permutation mentioned in Section 2.3.

For proving lower bounds in the black & white pebble game it turns out to be convenient to consider the pebbling strategy to be a sequence of moves where a move can be a placement or a removal of a pebble. The moves are regarded to be numbered in sequence. We say that the move whose number is z *happens* at time z. A vertex has a pebble (resp. is pebble-free) *at time* z if it has a pebble (resp. is pebble-free) after the z-th move. As long as we are careful to count only placements of pebbles when we count necessary moves, this concept of time in the end does not invalidate the fact that removals of pebbles do not take time. We will follow this approach in all our lower bound proofs that consider black and white pebbles. (In order to distinguish this slightly different concept of time from the concept of time introduced in the pebbling rules (i)–(iii), we will be denote move numbers with the letter z—for the German word "Zeit" meaning time. Intervals of moves will be denoted by Z.)

Theorem 2.4.3: Pebbling the bit reversal graph on N elements with $S \geq 2$ black and white pebbles takes at least time

$$T > \frac{N^2}{36S^2} + N.$$

Proof: For $S > N/6$ the theorem holds trivially. Thus let $S \leq N/6$.

Let s be the integer such that

$$3S \leq 2^s < 6S.$$

As in Theorem 2.3.2 the output path is considered as divided into $2^{n-s} > N/6S$ intervals I_j $(0 \leq j < 2^{n-s})$, of length 2^s. The argument given in the proof of Theorem 2.3.2 now has to be modified, however, since the intervals do not have to be pebbled in sequence.

Define $z_0 := 0$. Let the set Σ_0 be the empty set of intervals. For $1 \leq i \leq \lceil N/6S^2 \rceil$ inductively define z_i to be the first time after z_{i-1} at which any interval that is not in Σ_{i-1} has been pebbled and unpebbled completely. Denote this interval by I_{j_i}. At z_i a pebble is removed from I_{j_i}, and at most $S-1$ other intervals have pebbles on them. Add these intervals and I_{j_i} to Σ_{i-1} to define the set Σ_i. Note that Σ_i has at most iS elements, and thus for $i \leq \lceil N/6S^2 \rceil$ the interval I_{j_i} exists.

Analogously to the proof of Theorem 2.3.2 we will now argue that between z_{i-1} and and z_i more than $N/6$ placements have to occur. We start by observing that at time z_{i-1} the interval I_{j_i} is pebble-free and thus all of I_{j_i} has to be pebbled and unpebbled between z_{i-1} and z_i. The immediate predecessors of the 2^e vertices in I_{j_i} on the input path divide the input path canonically into $2^e - 1$ intervals of length 2^{n-e}. All but $S-1$ of these intervals are pebble-free at time z_{i-1} and all but $S-1$ (different) intervals are pebble-free at time z_i. Thus at least $2^e - (2S-2) > S$ intervals on the input path are pebble-free both at z_{i-1} and at z_i. All these intervals have to be pebbled and unpebbled completely between z_{i-1} and z_i. This takes at least $S \cdot 2^{n-e} > N/6$ placements on the input path. Thus before $z_{\lceil N/6S^2 \rceil}$ more than

$$\left\lceil \frac{N}{6S^2} \right\rceil \frac{N}{6} \geq \frac{N^2}{36S^2}$$

placements on the input path have to occur. At least N more placements occur on the output path. ∎

The improvement of the time-space tradeoff for the bit reversal graph by using black and white pebbles relies heavily on a certain regularity of the bit reversal permutation that allows us to pebble certain chunks on the output path with only small modifications on the input path. It is our conjecture that there are permutations that do not exhibit any regularity of this kind and for which the time-space tradeoff for black and white pebbles just as for black pebbles has the form

$$ST = \Theta(N^2).$$

3 SUPERCONCENTRATORS

3.1 Introduction

Proving Conjecture C means finding graphs of arbitrarily large size that are very hard to pebble. As pointed out in Chapter 1, people have been looking for such graphs for quite some time. Even though nobody was able to approach superpolynomial lower bounds some graph families have been studied that are fairly hard to pebble (see [Rei78], [To78] and Chapter 2 of this thesis.) A natural way of constructing graphs with even more dramatic time-space tradeoffs is to select one of those graph families and use its graphs as basic building blocks in a construction scheme that connects them in an appropriate way to amplify their bad properties. For reasons that will become apparent later on, superconcentrators are a suitable graph family for this purpose.

Definition 3.1.1: A directed acyclic graph C with bounded in-degree, N inputs and N outputs is called an *N-superconcentrator* if for every k such that $1 \leq k \leq N$ and for every pair of subsets V_1 of k inputs and V_2 of k outputs there are k vertex-disjoint paths connecting the vertices in V_1 to the vertices in V_2.

Note that we do not assume the ability to say which input is connected to which output.

Definition 3.1.1 shows that superconcentrators have to be fairly dense graphs in order to be able to achieve the routing necessary to join inputs to outputs in all required ways. It is therefore interesting to find out how many edges are necessary to build N-superconcentrators. It is relatively easy to construct N-superconcentrators with a maximum in-degree of 2, a depth (i.e. length of the longest path) of $O(\log N)$ and $O(N \log N)$ edges (and vertices). Figure 3 shows such an N-superconcentrator for $N = 16$. It is constructed by putting two FFT-graphs back to back and fulfills the even stronger property that we are able to specify beforehand which inputs have to be connected to which outputs. (Such graphs are called *connectors*.)

Valiant shows in [Va76] that N-superconcentrators exist that have only $O(N)$ edges. We will call such superconcentrators *linear*. Valiant bases his result on a paper by Pinsker ([Pin73]). Pippenger ([Pip77]) gives an improved construction of linear superconcentrators. His N-superconcentrators have a maximum in-degree of 9, a depth of $O(\log N)$ and at most $40N$ edges (and vertices). However, his construction involves a step that is based on a probabilistic counting argument. Recently Gabber & Galil ([GG79]) explicitly constructed linear superconcentrators.

Interest in superconcentrators first arose in the context of telephone switching networks. Then superconcentrators were found in graphs that represent practical algorithms like the multiplication of an N-vector by a non-singular $N \times N$ matrix (see [Va76]). The conjecture that no linear superconcentrators exist gave hope towards showing nonlinear lower bounds on the complexity of such algorithms. Valiant's result shows that superconcentrators cannot be applied in this way to show nonlinear lower bounds. However, it provides a family of highly interconnected sparse graphs. Such graphs are very good candidates for inducing dramatic time-space tradeoffs.

As mentioned in Chapter 1 we will include a treatment of the black & white pebble game in our results, since it is a straightforward generalization of the black pebble game. Specifically we will show all lower bounds using black and white pebbles and all upper bounds using black pebbles. Since the bounds will match each other asymptotically, this proves asymptotically equal time-space tradeoffs for both the black and the black & white pebble game.

Section 3.2 discusses a lower bound and Section 3.3 discusses an upper bound on the time-space tradeoff for pebbling superconcentrators.

3.2 The Lower Bound in the Black & White Pebble Game

In [To78] Tompa shows a lemma that he uses to prove lower bounds on pebbling superconcentrators. We generalize his lemma to the black & white pebble game. Let us say that we pebble r outputs of an N-superconcentrator in a time interval Z if either Z contains r moves that pebble outputs of the superconcentrator or $r \geq N$ and at the end of Z all outputs of the superconcentrator are pebbled.

Lemma 3.2.1 (Basic Lower Bound Argument, BLBA): In order to pebble $S_b + S_e + 1$ outputs of an N-superconcentrator starting with a configuration of at most S_b black and white pebbles on the graph and finishing with a configuration of at most S_e black and white pebbles on the graph, at least $N - S_b - S_e$ inputs of the graph have to be pebbled and unpebbled.

Proof: The proof is indirect. Assume that there are $S_b + S_e + 1$ outputs that can be pebbled starting with S_b pebbles and finishing with S_e pebbles without both pebbling and unpebbling any of $S_b + S_e + 1$ inputs. Since the graph is a superconcentrator, there are $S_b + S_e + 1$ vertex-disjoint paths connecting these inputs to the outputs to be pebbled. At least one of these paths starts out and ends up pebble-free. Its output has to be pebbled. Since the path ends up pebble-free its input has to be pebbled and unpebbled. This is a contradiction. ∎

Corollary 3.2.2: Pebbling an N-superconcentrator with S black and white pebbles takes at least $\Omega(N^2/S)$ pebblings of the inputs.

Proof: Iterate the BLBA $\lfloor N/(2S+1) \rfloor$ times. ∎

By Corollary 3.2.2 it is asymptotically at least as hard to pebble superconcentrators as it is to pebble bit reversal graphs. Thus superconcentrators are a graph family with a rather dramatic time-space tradeoff. However, this fact alone does not make superconcentrators good building blocks for constructing bad graphs. Bit reversal graphs for instance are not suited for this purpose. The reason why superconcentrators are an appropriate family to use in the construction of bad graphs lies in the existence of the Basic Lower Bound Argument. Such an argument does not hold for bit reversal graphs. The BLBA holds for superconcentrators because their inputs (resp. outputs) are completely symmetric to each other and thus indistinguishable. It is this symmetry and in particular its formulation through the BLBA that we apparently have to exploit, if we want to achieve superpolynomial lower bounds on pebbling times.

3.3 The Upper Bound in the Black Pebble Game

In this section we consider special classes of superconcentrators that can be pebbled efficiently.

There are classes of superconcentrators for which the lower bound proved in the last section is tight up to a constant factor. An example of such a class is the class of superconcentrators constructed by putting two FFT-graphs back to back (see Figure 3), as can be proved by an argument similar to the one given in [SS77]. However, as of now a similarly efficient pebbling strategy for linear superconcentrators is not known.

In [Pip77] Pippenger gives a recursive construction that he uses to prove the existence of linear superconcentrators. Recently Gabber and Galil combined Pippenger's construction with ideas of Margulis ([Ma73]) to explicitly construct linear superconcentrators. Before giving Pippenger's construction we have to introduce another graph concept.

Definition 3.3.1: Let n, κ, θ_1 and θ_2 be positive integers such that $\theta_1 < \theta_2$ and let $\theta_1 \mid \theta_2\kappa$. Let $\kappa' = \theta_2\kappa/\theta_1$. An $(n,\kappa,\theta_1,\theta_2)$-*linear concentrator* is a bipartite graph with n left and $\theta_1\lceil n/\theta_2 \rceil$ right vertices such that each left vertex has a degree of at most κ and each right vertex has a degree of at most κ' and such that each subset X of left vertices with $|X| \leq n/2$ is connected to at least $|X|$ right vertices. $(\theta_1 < \theta_2)$ (An $(n,\kappa,\theta_1,\theta_2)$-linear concentrator has at most κn edges.)

Pippenger uses concentrators in order to construct linear superconcentrators in the following way:

Definition 3.3.2: Let $\lambda(N) = \theta_2\lceil N/\theta_1 \rceil$. An $(N, \kappa, \theta_1, \theta_2)$-*linear superconcentrator* is a linear N-superconcentrator that is recursively defined as follows:

(a) If $N \leq \theta_1$ then the $(N, \kappa, \theta_1, \theta_2)$-linear superconcentrator is the complete bipartite graph $K[N, N]$.

(b) If $N > \theta_1$ then the $(N, \kappa, \theta_1, \theta_2)$-linear superconcentrator has N inputs and N outputs such that the following holds.

Directed edges join the inputs with their corresponding outputs.

The inputs are also the left vertices of an $(N, \kappa, \theta_1, \theta_2)$-linear concentrator G_1. Edges in G_1 are directed from the left towards the right vertices.

The outputs are also the left vertices of an $(N, \kappa, \theta_1, \theta_2)$-linear concentrator G_2. Edges in G_2 are directed from the right towards the left vertices.

The right vertices of G_1 are also the inputs of a $(\lambda(N), \kappa, \theta_1, \theta_2)$-linear superconcentrator whose outputs are the right vertices of G_2.

(A schematic representation of this construction is given in Figure 4.)

Lemma 3.3.3: An $(N, \kappa, \theta_1, \theta_2)$-linear superconcentrator has at most

$$\left(\frac{2\kappa + 1}{1 - \theta_1/\theta_2}\right) N + O(\log N)$$

vertices and edges. Its depth is $O(\log N)$.

Proof: The proof is a straightforward induction on N. ∎

In [Pip77] Pippenger proves the existence of $(n, 6, 4, 6)$-linear concentrators and thus by the above definition also the existence of $(N, 6, 4, 6)$-linear superconcentrators. These superconcentrators have $39N + O(\log N)$ edges and Pippenger shows in addition that they have at most $40N$ edges.

Recently Gabber and Galil extended ideas of Margulis ([Ma73]) to explicitly construct $(n, 112, 16, 17)$-linear bounded concentrators. Their construction involves additional technical constraints ($\lceil n/\theta_2 \rceil$ has to be a perfect square) and leads to linear N-superconcentrators with $3825N + O(\sqrt{N})$ edges.

We will give a pebbling strategy that pebbles $(N, \kappa, \theta_1, \theta_2)$-linear superconcentrators using $\Omega(N)$ pebbles in time $O(N \cdot (N/S)^{\alpha})$ where $\alpha = 1 + 2\log_{\theta_2/\theta_1}\kappa$. Thus it pebbles Pippenger's ($(N, 6, 4, 6)$-linear) superconcentrators in time $O(N \cdot (N/S)^{0.84})$ and Gabber and Galil's ($(N, 112, 16, 17)$-linear) superconcentrators in time $O(N \cdot (N/S)^{156.67})$.

Let us denote the $(N, \kappa, \theta_1, \theta_2)$-linear superconcentrator with $C(N, \kappa, \theta_1, \theta_2)$. Imagine $C(N, \kappa, \theta_1, \theta_2)$ to be unfolded i times by applying the recursion in Definition 3.3.2. We then get a similar picture as in Figure 4, except now G_1 and G_2 are replaced by $G_{i,1}$ and $G_{i,2}$, which are concatenations of i bipartite graphs that become smaller towards the middle of the superconcentrator (see Figure 5). The superconcentrator in the middle of Figure 4 is now $C(\lambda^i(N), \kappa, \theta_1, \theta_2)$, where, as is easily proved inductively,

$$\lambda^i(N) \leq \left(\frac{\theta_1}{\theta_2}\right)^i N + \frac{\theta_1(\theta_2 - 1)}{\theta_2 - \theta_1}. \tag{1}$$

Assume that S pebbles are available for pebbling $C(N, \kappa, \theta_1, \theta_2)$, where $S \geq c \log N$ for a large enough constant $c > 0$. Unfold $C(N, \kappa, \theta_1, \theta_2)$ j times, where $j = j(N, S)$ is minimum such that $C(\lambda^j(N), \kappa, \theta_1, \theta_2)$ has at most S vertices. By the above estimate (1) for m and Lemma 3.3.3

$$j = \log_{\theta_2/\theta_1}\frac{N}{S} + O(1). \tag{2}$$

Furthermore note that $j(\lambda(N), S) = j(N, S) - 1$.

We can pebble any r outputs of $C(N, \kappa, \theta_1, \theta_2)$ with the following strategy.

C-PEBBLE(r, S):

If $j = 0$ then pebble $C(N, \kappa, \theta_1, \theta_2)$ in topological order.

If $j > 0$ then pebble $C(N, \kappa, \theta_1, \theta_2)$ in three phases:

1. Put pebbles on all inputs of $C(\lambda^j(N), \kappa, \theta_1, \theta_2)$.
2. Put pebbles on all outputs of $C(\lambda^j(N), \kappa, \theta_1, \theta_2)$.
3. Pebble the r outputs of $C(N, \kappa, \theta_1, \theta_2)$, but keep permanent pebbles on all outputs of $C(\lambda^j(N), \kappa, \theta_1, \theta_2)$.

Thus in phase 1 we put pebbles on all outputs of $G_{j,1}$, in phase 2 we put pebbles on all outputs of $C(\lambda^j(N),\kappa,\theta_1,\theta_2)$, and in phase 3 we pebble all outputs of $G_{j,2}$ while preserving the pebbles on the outputs of $C(\lambda^j(N),\kappa,\theta_1,\theta_2)$. Because of the choice of $j(N,S)$ we can pebble $C(\lambda^j(N),\kappa,\theta_1,\theta_2)$ in topological order and phase 2 takes time $O(S)$. Phase 1 takes as much time as it takes to pebble $G_{j,1}$.

Let $i \geq 0$ and if $i > 0$ then let i be such that $\lambda^{i-1}(N) > \theta_1$. (This ensures that the recursion in Definition 3.3.2 can be applied i times to $C(N,\kappa,\theta_1,\theta_2)$.) Let $C_i(N,\kappa,\theta_1,\theta_2)$ be the graph which is created from $C(N,\kappa,\theta_1,\theta_2)$ by deleting all the edges in the graph $C(\lambda^i(N),\kappa,\theta_1,\theta_2)$ that occurs in the middle of $C(N,\kappa,\theta_1,\theta_2)$. Note that if $i = 0$ then $C_i(N,\kappa,\theta_1,\theta_2)$ is the empty bipartite graph with N left and N right vertices, and if $i > 0$ then $C_i(N,\kappa,\theta_1,\theta_2)$ follows the same recursion as in Definition 3.3.2, except that the graph $C(\lambda(N),\kappa,\theta_1,\theta_2)$ is replaced by the graph $C_{i-1}(\lambda(N),\kappa,\theta_1,\theta_2)$. Figure 6 illustrates the graph $C_i(N,\kappa,\theta_1,\theta_2)$ for $i = 4$.

Because in phase 3 permanent pebbles are kept on the output vertices of $C(\lambda^j(N),\kappa,\theta_1,\theta_2)$, phase 3 takes at most as long as it takes to pebble r outputs of the graph $C_j(N,\kappa,\theta_1,\theta_2)$.

We will find upper bounds for the time needed for phases 1 and 3 by a relatively crude argument that applies to general directed acyclic graphs G with depth δ. Then we will apply this result to $G_{j,1}$ and $C_j(N,\kappa,\theta_1,\theta_2)$, both of which have a depth $\delta = O(\log N)$.

Let G be an acyclic graph with depth δ. We can pebble G by successively pebbling all its outputs. Let τ be an output of G. Pebbling τ means pebbling the graph $G(\tau)$ induced by all vertices from which τ is reachable. We can pebble $G(\tau)$ with $O(\delta)$ pebbles using the procedure DEPTH-FIRST-PEBBLE given in [PTC77]. In order to find an upper bound for the time needed to pebble $G(\tau)$ we investigate the following trees.

Definition 3.3.4: Let G be a directed acyclic graph with depth δ and a unique output vertex τ. The *unfolding* of G is a tree U_G of depth δ. Each vertex v' in U_G is the *image* of a vertex v in G. The tree U_G is the unique tree with the following properties.

(a) There is exactly one image of the output vertex τ in U_G and it is the root of U_G.

(b) If the vertex w' in U_G is the image of the vertex w in G, then w' has exactly one child v' in U_G for each vertex v in G such that (v,w) is an edge in G. The vertex v' is an image of the vertex v.

Fact 3.3.5: Each strategy for pebbling the root of U_G in time T canonically induces a strategy for pebbling the output τ in G in a time which does not exceed T. The strategy for pebbling G pebbles a vertex v whenever the strategy for pebbling U_G pebbles an image of v and at the same time no other image of v in U_G is pebbled. A pebble is taken off of v in G when a pebble is taken off of an image v' of v in U_G, and no other image of v in U_G is pebbled. Therefore a vertex v in G has a pebble exactly when one of its images in U_G has a pebble.

Fact 3.3.6: DEPTH-FIRST-PEBBLE pebbles any tree of depth δ in linear time using $O(\delta)$ pebbles. (See [PTC77].)

From the above facts we can infer that the size of U_G gives an upper bound for the time necessary for pebbling G with $O(\delta)$ pebbles. Let us first bound the size of the unfoldings of the graphs $G_{j,1}(\tau)$ for each output τ of $G_{j,1}$.

Lemma 3.3.7: For each output τ of $G_{j,1}$ the unfolding of $G_{j,1}(\tau)$ has $O((N/S)^{\beta+1})$ vertices, where $\beta = \log_{\theta_2/\theta_1}\kappa$.

Proof: By equation (2) the graph $G_{j,1}$ has a depth of $j = \log_{\theta_2/\theta_1} N/S + O(1)$. Furthermore it has a maximum in-degree of κ'. Thus for each output τ of $G_{j,1}$ the unfolding of $G_{j,1}(\tau)$ has $O(\kappa'^j) = O((N/S)^{\beta+1})$ vertices. ∎

Corollary 3.3.8: Phase 1 of C-PEBBLE takes time $O(N \cdot (N/S)^{\beta})$.

Proof: By the choice of j the graph $G_{j,1}$ has at most $S/2$ output vertices. Since $S \geq c \log N$ for a sufficiently large constant $c > 0$, Fact 3.3.5, Fact 3.3.6, and Lemma 3.3.7 imply that $G_{j,1}(\tau)$ can be pebbled in with $S/2$ pebbles in time $O((N/S)^{\beta+1})$ for all outputs τ of $G_{j,1}$. Thus pebbles can be put on all outputs of $G_{j,1}$ in time $O(S \cdot (N/S)^{\beta+1}) = O(N \cdot (N/S)^{\beta})$. ∎

Corollary 3.3.8 gives us the upper bound for the time that phase 1 takes. As we already mentioned phase 2 takes time $O(S)$. We will now find an upper bound on the time for phase 3.

Lemma 3.3.9: Let $L(i)$ (resp. $V(i)$) be the maximum number of leaves (resp. vertices) in the unfolding of $C_i(N, \kappa, \theta_1, \theta_2)(\tau)$ for any output τ of $C_i(N, \kappa, \theta_1, \theta_2)$. Then $L(0) = V(0) = 1$. For $i > 0$ the following recurrences hold.

$$L(i) \leq 1 + \kappa\kappa' L(i-1)$$
$$V(i) \leq \kappa V(i-1) + \kappa\kappa' L(i-1) + 2$$

Proof: The case $i = 0$ is trivial. Assume $i > 0$. In order to bound the size of the unfolding of $C_i(N, \kappa, \theta_1, \theta_2)(\tau)$ we have to trace all possible paths backwards from τ in $C_i(N, \kappa, \theta_1, \theta_2)$ and to bound the multiplicities introduced by the in-degrees of vertices on the different levels.

For $L(i)$ one leaf is contributed by the input of $C_i(N, \kappa, \theta_1, \theta_2)$ which corresponds to τ. The other leaves result from the three maximum multiplicities introduced by G_2 (at most κ leaves), by $C_{i-1}(\lambda(N), \kappa, \theta_1, \theta_2)$ (at most $L(i-1)$ leaves) and by G_1 (at most κ' leaves).

For $V(i)$ two vertices are contributed by τ and its corresponding input of $C_i(N, \kappa, \theta_1, \theta_2)$. Since G_2 has an in-degree of at most κ, τ is connected to at most κ outputs of $C_{i-1}(\lambda(N), \kappa, \theta_1, \theta_2)$. Each of these outputs contributes a tree with at most $V(i-1)$ vertices and at most $L(i-1)$ leaves. Since the maximum in-degree of G_1 is κ' each leaf of $C_{i-1}(\lambda(N), \kappa, \theta_1, \theta_2)$ is connected to at most κ' inputs of $C_j(N, \kappa, \theta_1, \theta_2)$. Therefore there are at most

$$\kappa\big(V(i-1) + \kappa' L(i-1)\big) + 2$$

vertices in the unfolding of $C_i(N, \kappa, \theta_1, \theta_2)(\tau)$, which proves the lemma. ∎

In order to solve these recurrences we give the following general theorem.

<u>Theorem 3.3.10</u>: Let a, b and c be non-negative real constants. The recurrence

$$T(0) = O(1)$$
$$T(i) \leq a\,T(i-1) + c\,b^i \qquad \text{if } i > 0$$

has the solution

$$T(i) = O(b^i) \qquad \text{if } a < b$$
$$T(i) = O(i\,b^i) \qquad \text{if } a = b$$
$$T(i) = O(a^i) \qquad \text{if } a > b.$$

Proof: It can easily be proved inductively that

$$V(i) = a^i V(0) + c\,b^i \sum_{0 \leq \nu \leq i-1} \left(\frac{a}{b}\right)^{\nu}. \tag{3}$$

If $a > b$ then both terms in (3) exhibit equal growth and $V(i) = O(a^i)$.

If $a = b$ then the second term in (3) dominates $V(i)$ and $V(i) = O(i\,a^i)$.

If $a < b$ then the second term in (3) dominates $V(i)$ and $V(i) = O(b^i)$. ∎

The recurrences in Lemma 3.3.9 are now easily solved.

Theorem 3.3.11: Both $L(i)$ and $V(i)$ exhibit an asymptotic growth of $O((\kappa\kappa')^i)$.

Proof: Substituting $a = \kappa\kappa'$, $b = 1$, and $c = 1$ in Theorem 3.3.10 yields $L(i) = O((\kappa\kappa')^i)$.

Substituting $a = \kappa$, $b = \kappa\kappa'$ and a large enough constant $c > 0$ in Theorem 3.3.10 then yields $V(i) = O((\kappa\kappa')^i)$. ∎

Theorem 3.3.12: Any r outputs of the graph $C_j(N, \kappa, \theta_1, \theta_2)$ can be pebbled in time $O(r \cdot (N/S)^a)$.

Proof: By definition of $V(i)$, the estimate (2) for j and Theorem 3.3.11 the size of the unfolding of $C_j(N, \kappa, \theta_1, \theta_2)(\tau)$ for any output τ of $C_j(N, \kappa, \theta_1, \theta_2)$ is $O((\kappa\kappa')^j) = O((\kappa\kappa')^{\log_{\theta_2/\theta_1} N/S}) = O((N/S)^a)$. For pebbling each $C_j(N, \kappa, \theta_1, \theta_2)(\tau)$ we have $S/2$ pebbles available. Since $S \geq c \log N$ for a sufficiently large constant $c > 0$ Facts 3.3.5 and 3.3.6 imply that we can pebble each output of $C_j(N, \kappa, \theta_1, \theta_2)$ in time $O((N/S)^a)$. In total r outputs have to be pebbled. ∎

Theorem 3.3.13: The strategy C-PEBBLE pebbles any r outputs of $C_j(N, \kappa, \theta_1, \theta_2)$ in time

$$T \leq O(N \cdot (N/S)^{\log_{\theta_2/\theta_1} \kappa}) + O(r \cdot (N/S)^{1+2\log_{\theta_2/\theta_1} \kappa}).$$

Proof: The first term corresponds to the duration of phase 1 and the third term corresponds to the duration of phase 3. Phase 2 is always dominated by phase 1. ∎

Corollary 3.3.14: The strategy C-PEBBLE pebbles the linear superconcentrator $C(N, \kappa, \theta_1, \theta_2)$ in time $O(N \cdot (N/S)^a)$.

Proof: For $r = N$ phase 3 dominates the pebbling time. ∎

If we compare the upper bound of Corollary 3.3.14 with the lower bound of Theorem 3.2.2 we detect a difference of 8.84 in the exponent (for Pippenger's superconcentrators). This is a quite considerable gap for small S. For $S = \Theta(N/\log N)$ the upper bound of Theorem 3.3.11 implies, however, that $T = O(N(\log N)^{9.84}) = o(N^{1+\epsilon})$ for any $\epsilon > 0$. The BLBA gives in this case $T = O(N \log N)$ and the gap between the bounds is relatively small, namely a factor of $O((\log N)^{8.84})$.

4 STACKS OF SUPERCONCENTRATORS

4.1 Introduction

In the last chapter we decided to use superconcentrators as the basis for the construction of bad graphs. What we still need is an appropriate scheme for connecting several superconcentrators in a way that amplifies their relevant properties. The most straightforward way of doing this is to stack several superconcentrators of the same size in series. This approach is also motivated by the results given in [PT77] and [Rei78].

It turns out that already this simple-minded connection scheme is enough to yield graphs for proving Conjecture C. The proof is, however, substantially more involved than the construction itself. We are here—as often in the theory of computation—confronted with an easy construction whose properties are hard to prove.

The graph family we will consider has two parameters and is defined as follows.

Definition 4.1.1: Let $n \geq 6$. For i such that $1 \leq i \leq k$ let C_i be a copy of Pippenger's $(n, \kappa, \theta_1, \theta_2)$-linear superconcentrator. Let $C(n,k)$ be the graph created by joining the outputs of C_i to the corresponding inputs of C_{i+1} with directed edges $(1 \leq i < k)$. The graph $C(n,k)$ has at least $2nk$ and at most $40nk$ and thus $\Theta(nk)$ vertices. (Figure 7 schematically shows $C(n,k)$.)

As we will prove in Section 4.2, as long as $S \leq n/20$, the task of pebbling $C(n,k)$ with S black and white pebbles takes time

$$T = n\left(\frac{nk}{64S}\right)^k.$$

Furthermore in Section 4.3 we will pebble Pippenger's superconcentrators using only black pebbles in time

$$T = n\,O(\frac{nk}{S})^{9.84k}$$

and even in linear time if $S \geq 40n$.

Now let $S = \Omega(N/\log N)$. Choosing $n = 6S$ and $k = \lfloor N/S \rfloor$ we get a graph $C(n,k)$ of size $\Theta(N)$ such that pebbling $C(n,k)$ with S black and white pebbles takes time

$$T = S\ \Omega(\frac{N}{S})^{\Omega(\frac{N}{S})} \tag{1}$$

(It is important to notice that the graph $C(n,k)$ satisfying this lower bound depends on S. There is apparently no single graph that is bad for every S. Many graphs contribute single data points to the enveloping lower bound curve.)

Moreover each graph $C(n,k)$ of size N $(2nk \leq N \leq 40nk)$ can be pebbled using only black pebbles in time

$$T \leq \frac{N}{2k}\ O(\frac{N}{S})^{O(k)} \leq S\ O(\frac{N}{S})^{O(k)}$$

and even in linear time if $k \geq 20N/S$. Thus each $C(n,k)$ of size N can be pebbled in time

$$T = S\ O(\frac{N}{S})^{O(\frac{N}{S})}. \tag{2}$$

This shows that for both the black and the black & white pebble game, the family of graphs $C(n,k)$ has a time-space tradeoff of the form

$$T = S\ \Theta(\frac{N}{S})^{\Theta(\frac{N}{S})}.$$

Analysis of this formula shows that if $S = O(N/\log N)$ then $T = N^{\Omega(\log\log N)}$. Furthermore T is superpolynomial as long as $S \leq cN \log\log N/\log N$ for a suitably small constant $c > 0$.

Conjecture C is thus proved. Furthermore this time-space tradeoff implies a lower bound on S_J (see Chapter 1) of the form

$$S_J = \Omega(N \log\log N/\log N).$$

This lower bound does not match the upper bound on S_J that follows from Reischuk's studies ([Rei78]) and has the form.

$$S_J = o(N/\log\log\log N).$$

In Chapter 5 we will concentrate on closing the gap between these two bounds.

Note that the asymptotic lower bound (1) discussed in Section 4.2 can (with different constants) also be obtained for stacks of any linear superconcentrators. The asymptotic upper bound (2) discussed in Section 4.3 holds (with different constants) for all $(n, \kappa, \theta_1, \theta_2)$-linear superconcentrators. This in particular implies that there are constructible graph families that realize the time-space tradeoffs discussed in this chapter.

4.2 The Lower Bound in the Black & White Pebble Game

We can iterate the Basic Lower Bound Argument (Lemma 3.2.1) to find a lower bound on the time-space tradeoff for pebbling $C(n,k)$.

Theorem 4.2.1: In order to pebble all outputs of $C(n,k)$ using S black and white pebbles ($2 \leq S \leq (n-1)/4$) (starting with any configuration of pebbles on the graph) we need T placements such that

$$T \geq n\left(\frac{n}{10S}\right)^k.$$

Proof: The subgraph C_k together with the outputs of C_{k-1} and the edges joining C_{k-1} with C_k is an n-superconcentrator. Thus we can apply the BLBA $\lfloor \frac{n}{2S+1} \rfloor$ times to prove that

$$(n-2S)\left\lfloor \frac{n}{2S+1} \right\rfloor$$

placements of pebbles on outputs of C_{k-1} are necessary to pebble all outputs of C_k. Iterating this argument through $C_{k-1}, \ldots, C_1$ we find that

$$(n-2S)\left\lfloor \frac{n-2S}{2S+1} \right\rfloor^{k-1}\left\lfloor \frac{n}{2S+1} \right\rfloor \geq \frac{n}{2}\left(\frac{n}{10S}\right)^{k-1}\left(\frac{n}{5S}\right) = n\left(\frac{n}{10S}\right)^k$$

inputs of C_1 have to be pebbled. (Observe that $n-2S \geq 2S+1$ since $S \leq (n-1)/4$.) ∎

If the above theorem would already yield a superpolynomial growth of T for $S = O(N/\log N)$ then we would have proved Conjecture C using a simple-minded connection scheme (stacking) and a simple-minded proof (iteration of the BLBA). This cannot be expected and indeed is not the case. But the argument given in the proof of Theorem 4.2.1 can be improved considerably. This is because we did

not take into account at all how the pebbles are distributed over $C(n,k)$. Let us call a pebbling strategy *fair* if it distributes the S pebbles evenly over $C(n,k)$ and only assigns $\lfloor S/k \rfloor$ pebbles to the superconcentrator C_i $(1 \leq i \leq k)$. In this case the argument given in Theorem 4.2.1 should go through, even if we substitute for S the quantity $\lfloor S/k \rfloor$. We would then get a lower bound of the form

$$T \geq n\Omega(\frac{nk}{S})^k.$$

Of course there may be many strategies that are not fair in this sense but concentrate great numbers of pebbles on different levels at different times. However, this means that on other levels of $C(n,k)$ there will be fewer pebbles available at those times. If we analyze these interdependencies accurately enough we will be able to tighten the bound given in Theorem 4.2.1 by substituting tighter estimates for the number of (locally available) pebbles in the individual applications of the BLBA in the proof of Theorem 4.2.1.

Again we will use the approach of considering removals as well as placements of pebbles as moves in the pebble game (see remarks in Section 2.4). Since the BLBA only counts placements this will not invalidate our results.

Let us start by considering the outputs of C_k as numbered in the order in which they are (first) pebbled. Let z_i be the time at which output i is pebbled $(1 \leq i \leq n,\ z_0 := 0,\ z_{n+1} :=$ number of the last move of the strategy). Let $[z', z'']$ be the interval starting with move z' and ending with move z'' inclusively. Let p_i be the minimum number of pebbles on C_k after any of the moves in $[z_{i-1}, z_i]$ $(1 \leq i \leq n,\ p_{n+1} := 0)$. Observe that $p_i \leq S$ for $1 \leq i \leq n+1$. We will consider disjoint intervals of numbers $[i,j] \subset [1,n]$ which will represent disjoint time intervals $[z'_i, z''_j]$ where z'_i and z''_j $(z_{i-1} \leq z''_{i-1} < z'_i \leq z_i)$ for $0 \leq i \leq n$ are times that will be specified later. If convenient we will not explicitly distinguish between $[i,j]$ and $[z'_i, z''_j]$.

The disjoint intervals $[i,j]$ will be chosen such that the BLBA can be applied on C_k to each of them. They will generally be of different lengths depending on the maximum number of pebbles on C_k during an interval. The objective is to find a large number of intervals to which we can apply the BLBA with very tight space estimates.

The applications of the BLBA to the intervals will yield information about how many outputs of C_{k-1} have to be pebbled and how many pebbles are available to do this. Thus we will be able to give a recursive relationship between the time necessary to pebble the outputs of C_k and the time necessary to pebble the outputs of C_{k-1}. Solving this recurrence completes the proof.

For this program to be realizable the intervals $[i,j]$ have to be "good" acording to the following definition.

Definition 4.2.2: An interval $[i,j] \subset [1,n]$ is called *good* if it fulfills the following three requirements:

$$p_i \leq \frac{j-i}{2} \tag{3}$$

$$p_{j+1} \leq \frac{j-i}{2} \tag{4}$$

$$p_k > \frac{j-i}{8} \qquad \text{for } i < k \leq j. \tag{5}$$

Note that the length of each good interval is $j-i+1 \leq 8S$. Good intervals are important because of the following lemma.

Lemma 4.2.3: During the good interval $[i,j]$ at least $n-2S$ outputs of C_{k-1} are pebbled.Only $S-1-\lfloor \frac{j-i}{8} \rfloor$ pebbles are available for doing this.

Proof: Assume that the interval $[i,j]$ is good in the above sense. Then because of (3) there is a latest time z' $(z_{i-1} \leq z' < z_i)$ such that at most $(j-i)/2$ pebbles are on C_k at z'. (Observe that move z_i places a pebble on C_k and therefore at time $z_i - 1$ there are fewer pebbles on C_k than at time z_i.) Let x'_i be the number of pebbles on C_k at z'. Define $z'_i := z' + 1$.

Furthermore, because of (4) there is an earliest time z''_j $(z_j \leq z''_j < z_{j+1})$ such that at most $(j-i)/2$ pebbles are on C_k at time z''_j $(z''_n := z_{n+1})$. Let x''_j be the number of pebbles on C_k at z''_j. (Observe that for $1 < i \leq n$ we have $z''_{i-1} < z'_i$ and thus $[z'_{i_1}, z''_{j_1}]$ and $[z'_{i_2}, z''_{j_2}]$ are disjoint if $[i_1,j_1]$ and $[i_2,j_2]$ are disjoint.) During $[z'_i, z''_j]$, exactly $j-i+1$ outputs of C_k are pebbled starting with a configuration of x'_i and ending with a configuration of x''_j pebbles on C_k. Because of (3) and (4) the BLBA can be applied. The application yields that at least $n-x'_i-x''_j$ inputs of C_k have to be pebbled and unpebbled during $[z'_i, z''_j]$. Furthermore at z'_i there are at most $S-x'_i$ pebbles on $C_1,\ldots,C_{k-1}$, and at time z''_j there are at most $S-x''_j$ pebbles on $C_1,\ldots,C_{k-1}$. Therefore at least

$$n-x'_i-x''_j-(S-x'_i)-(S-x''_j) = n-2S$$

outputs of C_{k-1} have to be pebbled during $[z'_i, z''_j]$.

Furthermore, because of (5), at all times during $[z'_i, z''_j]$ more than $(j-i)/8$ pebbles stay on C_k and therefore at most $S-1-\lfloor\frac{j-i}{8}\rfloor$ pebbles are available for pebbling $C_1,\ldots,C_{k-1}$. Since move z'_i places a pebble on C_k, during $[z'_i+1, z''_j]$ at least $n-2S$ outputs of C_{k-1} have to be pebbled using at most $S-1-\lfloor\frac{j-i}{8}\rfloor$ pebbles. ∎

Lemma 4.2.3 shows how we can apply the BLBA to a good interval and proceed inductively on k. The following purely combinatorial lemma provides us with the necessary statement about the abundance of good intervals.

Lemma 4.2.4: Let $r \leq n$. We can find a set of disjoint good intervals in $[1,r]$ that covers at least $\frac{r}{4}-S-p_{r+1}$ elements of $[1,r]$.

Proof: By induction on r.

If $r < 4S$ then the statement of the lemma is trivial. Thus let $r \geq 4S$.

Let i be maximum such that $p_i \leq \frac{r+1-i}{4}$. (Such an i exists because $i=1$ is a candidate.) For k such that $i < k \leq i+\lfloor\frac{r+1-i}{2}\rfloor$ we have

$$p_k > \frac{r+1-k}{4} \geq \frac{r+1-i-\left\lfloor\frac{r+1-i}{2}\right\rfloor}{4} \geq \frac{r+1-i}{8}.$$

Also since p_i is integer

$$p_i \leq \frac{1}{2}\left\lfloor\frac{r+1-i}{2}\right\rfloor.$$

We can inductively assume that the number of elements in $[1,i-1]$ that can be covered by disjoint good intervals is at least

$$\frac{i-1}{4}-S-p_i \geq \frac{i-1}{4}-S-\frac{1}{2}\left\lfloor\frac{r+1-i}{2}\right\rfloor.$$

We have to make a case distinction.

Case 1: Assume there is a $j \in [i+\lfloor\frac{r+1-i}{2}\rfloor, r]$ such that $p_{j+1} \leq \lfloor\frac{j-i}{2}\rfloor$. Let j be chosen to be the smallest such point. Then $[i,j]$ is a good interval whose length is at least $\lfloor\frac{r+1-i}{2}\rfloor+1$ and thus at least

$$\frac{i-1}{4}-S-\frac{1}{2}\left\lfloor\frac{r+1-i}{2}\right\rfloor+\left\lfloor\frac{r+1-i}{2}\right\rfloor+1 \geq \frac{r}{4}-S \geq \frac{r}{4}-S-p_{r+1}$$

elements in $[1,r]$ are covered with disjoint good intervals.

Case 2: Otherwise we have $p_{r+1} > \lfloor \frac{r-i}{2} \rfloor$ and thus $p_{r+1} \geq \frac{r+1-i}{2}$. Thus

$$\frac{r}{4} - S - p_{r+1} \leq \frac{r}{4} - S - \frac{r+1-i}{2} \leq \frac{i-1}{4} - S - \frac{1}{2}\left\lfloor \frac{r+1-i}{2} \right\rfloor$$

and the lemma holds again. ∎

We will now use Lemma 4.2.3 and Lemma 4.2.4 to construct a recursive relationship for the time to pebble $C(n,k)$.

Theorem 4.2.5: Let $T(n,k,S)$ be the time necessary to pebble $\lceil 9n/10 \rceil$ outputs of $C(n,k)$ with $S \leq n/20$ pebbles. Then

$$T(n,1,S) \geq \frac{n^2}{10S}, \tag{6}$$

$$T(n,k,S) \geq \min_{(x_1,\ldots,x_m)\in D} \sum_{1\leq i\leq m} T\left(n, k-1, S-1-\left\lfloor \frac{x_i-1}{8} \right\rfloor\right) \quad \text{for } k>1, \tag{7}$$

where D is an index set that contains all the ways in which we can select a large number of good intervals. Specifically

$$D = \{(x_1,\ldots,x_m) \mid m > \frac{n}{64S} \text{ and } 1 \leq x_i \leq 8S-6 \text{ for } 1 \leq i \leq m \text{ and } \sum_{1\leq i\leq m} x_i \geq \frac{n}{8} \}.$$

Proof: By induction on k.

$k = 1$: follows trivially as in Theorem 4.2.1.

$k > 1$: Assume any strategy for pebbling $C(n,k)$. Let $r = \lceil 9n/10 \rceil$ and let $S = n/20$. By Lemma 4.2.3 during a good interval of length x at least $n - 2S \geq r$ outputs of C_{k-1} have to be pebbled using at most $S - 1 - \lfloor \frac{x-1}{8} \rfloor$ pebbles. (Unless $x \leq 8S-6$ no pebbles are left for pebbling the outputs of C_{k-1}.) Inductively this takes at least

$$T\left(n, k-1, S-1-\left\lfloor \frac{x-1}{8} \right\rfloor\right)$$

steps. By Lemma 4.2.4 the total length of the disjoint good intervals we can find is at least

$$\frac{r}{4} - 2S \geq \frac{n}{8}.$$

Thus if we assume that we have m good interval with lengths x_i $(1 \leq i \leq m)$ and minimize over all possible choices of the intervals, we get the formula given in the theorem. ∎

All that is left in order to find a lower bound on the time-space tradeoff for pebbling $C(n,k)$ is to solve the above recurrence. This can be done using standard methods of calculus.

Our motivating discussion leads us to guess that $T(n,k,S) \geq f(n,k,S)$ where

$$f(n,k,S) = n\left(\frac{nk}{cS}\right)^k \tag{8}$$

and $c \geq 10$ is an appropriate constant, which turns out to be 64. This guess can be verified inductively.

Theorem 4.2.6: If $S \leq n/20$ then

$$T(n,k,S) \geq n\left(\frac{nk}{64S}\right)^k. \tag{9}$$

Proof: The theorem is obviously true if $k = 1$. Thus assume that $k > 1$.

By the inductive hypothesis we have after eliminating the floor-function and substituting real variables $y_i = x_i/8$ (note that $f(n,k,S)$ is decreasing in S for $S > 0$)

$$T(n,k,S) \geq \min_{(y_1,\ldots,y_m)\in D'} \sum_{1\leq i\leq m} f(n,k-1,S-y_i) \tag{10}$$

where D' is the set

$$D' = \{(y_1,\ldots,y_m) \mid m > \frac{n}{64S} \text{ and } 0 < y_i < S \text{ for } 1 \leq i \leq m \text{ and } \sum_{1\leq i\leq m} y_i \geq \frac{n}{64} \}.$$

Let us first assume that m is fixed. The expression $f(n,k-1,S-y)$ is a convex function in y for $0 < y < S$ (its second derivative is non-negative). Thus we have

$$\sum_{1\leq i\leq m} f(n,k-1,S-y_i) \geq m f\left(n,k-1,S-\frac{1}{m}\sum_{1\leq i\leq m} y_i\right).$$

Since $f(n,k-1,S-y)$ is also increasing in the range $0<y<S$ we have

$$m f\left(n,k-1,S-\frac{1}{m}\sum_{1\le i\le m} y_i\right) \ge m f\left(n,k-1,S-\frac{n}{64m}\right). \qquad (11)$$

The value of the right hand side of (11) can now be minimized with respect to m by differentiation. The minimum for $m > n/64S$ occurs at

$$m = \frac{nk}{64S}$$

and amounts to

$$n\left(\frac{nk}{64S}\right)^k.$$

This proves the theorem. ∎

4.3 The Upper Bound in the Black Pebble Game

The question arises, whether after the refinement of the argument used in the proof of Theorem 4.2.1 the lower bound (9) that has been obtained on the time-space tradeoff for pebbling $C(n,k)$ is indeed asymptotically tight. This is the case, and a matching upper bound is proved in this section. The pebbling strategy that we use to establish the upper bound is derived from the strategy C-PEBBLE for pebbling $(n,\kappa,\theta_1,\theta_2)$-linear superconcentrators (see Section 3.3).

Assume that $S \ge c_1 k \log n$ pebbles are given, where $c_1 > 0$ is a sufficiently large constant. We define the following fair strategy STACK-PEBBLE that pebbles r outputs of $C(n,k)$.

STACK-PEBBLE(r,k,S):

Permanently assign $\lfloor S/k \rfloor$ pebbles to C_k and the rest of the pebbles to $C_1,\ldots,C_{k-1}$. Apply C-PEBBLE$(r,\lfloor S/k \rfloor)$ to C_k to pebble r outputs of C_k with its $\lfloor S/k \rfloor$ pebbles. This requires pebbling a certain number r' of inputs of C_k and thus pebbling (at most) the same number of outputs of C_{k-1}. These outputs are pebbled by recursively applying STACK-PEBBLE$(r',k-1,S-\lfloor S/k \rfloor)$ to $C_1,\ldots,C_{k-1}$.

Theorem 4.3.1: STACK-PEBBLE pebbles any r outputs of $C(n,k)$ with S pebbles in time

$$T \leq r \sum_{1 \leq i \leq k} \left(\frac{c_2 nk}{S}\right)^{\alpha i} + n \sum_{0 \leq i \leq k-1} (k-i) \left(\frac{c_2 nk}{S}\right)^{\alpha i + \beta} \tag{12}$$

where $\alpha = 1 + 2\log_{\theta_2/\theta_1} \kappa$, $\beta = \log_{\theta_1/\theta_2} \kappa$, and $c_2 > 0$ is a sufficiently large constant.

Proof: In Section 3.3 it is shown that the pebbling strategy C-PEBBLE pebbles any r outputs of a $(n, \kappa, \theta_1, \theta_2)$-linear superconcentrator with $\lfloor S/k \rfloor$ pebbles in time

$$T \leq r \cdot (c_2 nk/S)^{\alpha} + n \cdot (c_2 nk/S)^{\beta} \tag{13}$$

where $c_2 > 0$ is a suitably large constant.

The proof of the theorem is by induction on k.

$k = 1$: See Theorem 3.3.13.

$k > 1$: We have

$$\left\lfloor \frac{S}{k} \right\rfloor \leq \left\lfloor \frac{S - \lfloor S/k \rfloor}{k-1} \right\rfloor.$$

Thus STACK-PEBBLE$(r', k-1, S - \lfloor S/k \rfloor)$ has at most

$$r' \sum_{1 \leq i \leq k-1} \left(\frac{c_2 nk}{S}\right)^{\alpha i} + n \sum_{0 \leq i \leq k-2} (k-1-i) \left(\frac{c_2 nk}{S}\right)^{\alpha i + \beta} \tag{14}$$

moves. By (13) we have

$$r' \leq r \cdot (c_2 nk/S)^{\alpha} + n \cdot (c_2 nk/S)^{\beta}. \tag{15}$$

Substituting (15) into (14) and adding the number of placements on C_k yields

$$T \leq \left(r\left(\frac{c_2 nk}{S}\right)^{\alpha} + n\left(\frac{c_2 nk}{S}\right)^{\beta}\right)\left(1 + \sum_{1 \leq i \leq k-1} \left(\frac{c_2 nk}{S}\right)^{\alpha i}\right)$$

$$+ n \sum_{0 \leq i \leq k-2} (k-1-i) \left(\frac{c_2 nk}{S}\right)^{\alpha i + \beta}$$

$$\leq r \sum_{1 \leq i \leq k} \left(\frac{c_2 nk}{S}\right)^{\alpha i} + n\left(\frac{c_2 nk}{S}\right)^{\beta} \sum_{0 \leq i \leq k-1} \left(\frac{c_2 nk}{S}\right)^{\alpha i}$$

$$+ n \sum_{0 \leq i \leq k-2} (k-1-i)\left(\frac{c_2 nk}{S}\right)^{\alpha i+\beta}$$

$$\leq r \sum_{1 \leq i \leq k} \left(\frac{c_2 nk}{S}\right)^{\alpha i} + n \sum_{0 \leq i \leq k-1} (k-i)\left(\frac{c_2 nk}{S}\right)^{\alpha i+\beta}. \quad \blacksquare$$

Corollary 4.3.2: STACK-PEBBLE pebbles $C(n,k)$ in time

$$T = n\, O(\frac{nk}{S})^{\alpha k}.$$

Proof: Since $\alpha \geq \beta + 1$, for $r = n$ the first term in (12) is dominating. ∎

5 THE GENERAL CASE

5.1 Introduction

The discussion of stacks of superconcentrators led to a proof of Conjecture C, and in addition to a lower bound on S_J of the form

$$S_J = \Omega(N \log\log N / \log N). \tag{1}$$

Reischuk's pebbling algorithm (see [Rei78]) implies an upper bound on S_J of the form

$$S_J = o(N / \log\log\log N). \tag{2}$$

In this Chapter we will locate S_J and show that

$$S_J = \Theta(N / \log\log N). \tag{3}$$

Thus neither bound (1) nor bound (2) is tight. Equation (3) can be inferred from the following time-space tradeoff for pebbling general directed acyclic graphs with black or with black and white pebbles.

$$T = S\ 2^{2^{\Theta(\frac{N}{S})}} \tag{4}$$

(Equation (4) says that all directed acyclic graphs with bounded in-degree can be pebbled with a sufficiently large number S of pebbles in time

$$T = S\ 2^{2^{O(\frac{N}{S})}}$$

and that there are graphs for which a time

$$T = S\ 2^{2^{\Omega(\frac{N}{S})}}$$

is necessary to pebble them with S pebbles.)

Moreover (4) implies that if $S = O(N/\log N)$ then $T = 2^{N^{\Omega(1)}}$, i.e., T is exponential in some positive power of N.

In the following sections we will prove (4). Section 5.2 discusses the upper bound part of (4) in the black pebble game. Section 5.3 shows the lower bound part of (4) using black and white pebbles.

5.2 The Upper Bound In The Black Pebble Game

In [PTC77] Paul, Tarjan, and Celoni give a recursive algorithm **BEST-PEBBLE** for pebbling any directed acyclic graph with a maximum in-degree d ($d \geq 2$) using $S \geq c_5 (d \log d)(N / \log N)$ pebbles ($c_5 > 0$ is a sufficiently large constant). They do not analyze the time efficiency of their algorithm, and in fact it may be quite inefficient. However, it is possible to modify their algorithm such that it makes efficient use of all S pebbles that are available. We call the modified algorithm **FAST-PEBBLE**. It is stated below, and its time analysis leads to an upper bound of the form

$$T \leq S\,(c_7 d)^{c_8^{(d+1)\frac{N}{S}}}$$

($c_7, c_8 > 1$ are suitably large constants.).

(Reischuk ([Rei78]) independently uses similar ideas to prove his result that

$$T \leq N\,2^{2^{O(f(d)\,\frac{N}{S}\log\frac{N}{S})}}$$

for $S = \Omega(N / \log^r N)$ where $r \in \mathbb{N}$. The function $f(d)$ is not further specified.)

Throughout this section we will use the sum of the number of vertices and the number of edges as a measure of the graph size.

Let G be a graph of size m with a maximum in-degree d ($d \geq 2$). The definition and analysis of the algorithm FAST-PEBBLE(G, S) that pebbles G with S pebbles involves a set of constants $c_1, \ldots, c_8 > 0$ on which a number of rather arbitrary-looking constraints have to be imposed. Some of these constraints are essential for **FAST-PEBBLE** to work properly; others are used in the time analysis of the algorithm. A list of the constraints and short explanations of the significance of the constants are contained in Appendix A. An example of a set of constants satifying all constraints is

$$c_1 = \frac{11}{50}, \quad c_2 = \frac{21}{200}, \quad c_3 = \frac{1}{5}, \quad c_4 = 5,$$
$$c_5 = 100, \quad c_6 = \frac{1}{343}, \quad c_7 = 29, \quad c_8 = 40.$$

Before we can define the algorithm **FAST-PEBBLE** we have to give the definition of a function which plays a central role in the following discussion.

Definition 5.2.1: Define

$$p := p(d, m, S) := \left\lfloor \frac{S^2(2c_2 - c_3) - (d+1)S}{dm} \right\rfloor - 1.$$

We will show that if G has size m and S is large enough, then we can pebble G with S pebbles using the following informally stated recursive algorithm:

FAST-PEBBLE(G, S):

If $m \leq S$ then pebble G in topological order.

If $m > S$ then partition G into two disjoint parts G_1 (of size m_1) and G_2 (of size m_2) such that no edges run from G_2 into G_1, and such that

$$\left\lceil \frac{m}{2} \right\rceil - \lfloor c_2 S \rfloor - d \leq m_1 \leq \left\lceil \frac{m}{2} \right\rceil - \lfloor c_2 S \rfloor.$$

(This partition can be found by starting with $G_1 = \emptyset$ and successively adding vertices to G_1 in topological order until G_1 has the desired size. Figure 8 illustrates the partition of G into G_1 and G_2.) Let E be the set of edges from G_1 into G_2.

Case 1 (Small Cut): If $|E| \leq \lfloor c_2 S \rfloor$ then partition the S pebbles into a set S_1 of size $\lceil (1 - c_1)S \rceil$ and a set S_2 of size $\lfloor c_1 S \rfloor$. Do **FAST-PEBBLE**($G_1, \lceil (1 - c_1)S \rceil$) using the pebbles in S_1; while doing this use the pebbles in S_2 to permanently pebble all sources of edges in E. Then take the pebbles in S_1 off G_1 and do **FAST-PEBBLE**($G_2, \lceil (1 - c_1)S \rceil$) using the pebbles in S_1.

Case 2 (Big Cut): If $|E| > \lfloor c_2 S \rfloor$ then partition the S pebbles into two sets S_1 (for use on G_1 only) and S_2 (for use on G_2 only) each of size $\lfloor \frac{S-dp}{2} \rfloor$ and a pool P of special pebbles of size dp (see Definition 5.2.1). Start doing **FAST-PEBBLE**($G_2, \lfloor \frac{S-dp}{2} \rfloor$) using the pebbles in S_2. When a situation occurs where the output of an edge in E has to be pebbled whose inputs in G_1 are not all pebbled, then temporarily suspend the pebbling of G_2. Do FAST-PEBBLE($G_1, \lfloor \frac{S-dp}{2} \rfloor$) using the pebbles in S_1 and leave the pebbles in P on all of the (at most dp) inputs of E that directly precede the p outputs of E that have to be pebbled next. Then continue pebbling G_2.

Reischuk ([Rei78]) independently uses essentially the same algorithm to prove his upper bound result. His parameters (especially the number of special pebbles) are different however, and as a consequence his bound is not tight enough to match the lower bound proved in the next section.

We will start the analysis of FAST-PEBBLE by making some remarks about notation.

Let FAST-PEBBLE be called with arguments (G, S) where G has size m. We say that FAST-PEBBLE is called on a problem of size (m, S). In the course of its execution FAST-PEBBLE calls itself recursively on the graphs G_1 and G_2. We will denote the corresponding problem sizes by (m', S'), so that m' is either m_1 or m_2 and S' is either $\lceil (1-c_1)S \rceil$ or $\lfloor \frac{S-dp}{2} \rfloor$.

Define $m_0 = c_4 d$. The number m_0 marks the threshold for the graph size above which FAST-PEBBLE will become non-trivial.

With two lemmas we will now prepare ourselves for proving the efficiency of FAST-PEBBLE.

Lemma 5.2.2: Let FAST-PEBBLE be called on an argument of size (m, S). Assume that $m \geq S, m \geq m_0$ and if Case 2 applies, that $p \geq 1$, i.e., the pool P of special pebbles is not empty. Then for all recursive calls to FAST-PEBBLE on subgraphs of size m' we have S' pebbles available such that $S' \geq 1$ and

$$\frac{m'}{S'} \leq \frac{m}{S} - c_3.$$

Proof: Cases 1 and 2 are handled separately.

Case 1: By definition of FAST-PEBBLE

$$\frac{m'}{S'} \leq \frac{m/2 + c_2 S + d}{(1-c_1)S}.$$

With constraints (A3), (A4) and (A5) (see Appendix A), using $m \geq m_0$ and $m \geq S$ it follows that

$$0 \leq (1-2c_1)m - \left(2c_3(1-c_1) + 2c_2\right)S - 2d,$$

i.e.,

$$\frac{m}{2} + c_2 S + d \leq \left(\frac{m}{S} - c_3\right)(1-c_1)S,$$

and thus

$$\frac{m'}{S'} \leq \frac{m}{S} - c_3.$$

Case 2: Because of constraint (A2) we have

$$m' \leq \frac{m}{2} - c_2 S + (d+1)$$

and

$$S' \geq \frac{S - dp - 1}{2}. \tag{5}$$

We always have $S' \geq 1$ since by constraint (A1)

$$\begin{aligned}
\frac{S^2(2c_2 - c_3) - (d+1)S}{m} &= S\left(\left(\frac{S}{m}\right)(2c_2 - c_3) - \frac{(d+1)}{m}\right) \\
&\leq S\left(2c_2 - c_3 - \frac{d+1}{m}\right) \\
&\leq S(2c_2 - c_3) \\
&< S
\end{aligned}$$

and by definition of p thus

$$dp < S - 2.$$

Substituting this into (5) yields $S' \geq 1$.

Now by definition of p again

$$p \leq \frac{S\big((2c_2 - c_3)S - (d+1)\big)}{dm} - 1.$$

This implies

$$d + 1 \leq (2c_2 - c_3)S - \left(\frac{m}{S}\right)(dp+1)$$

and therefore

$$m - 2c_2 S + (d+1) \leq m - \left(\frac{m}{S}\right)(dp+1) - c_3 S.$$

Because $p \geq 1$ we have

$$m - 2c_2 S + (d+1) \leq m - \left(\frac{m}{S}\right)(dp+1) - c_3(S - dp - 1)$$

i.e.

$$m - 2c_2 S + (d+1) \leq \left(\frac{m}{S} - c_3\right)(S - dp - 1)$$

which implies that

$$\frac{m'}{S'} \leq \frac{m - 2c_2 S + (d+1)}{S - dp - 1} \leq \frac{m}{S} - c_3. \qquad \blacksquare$$

Lemma 5.2.2 states that as we descend one level in the recursion, under certain circumstances there is a constant minimum decline in the quotient m/S of the problem size (m, S).

Lemma 5.2.3: Let (m, S) be such that $S \geq c_5 \log d\,(m/\log m)$. Then

(a) If $m \leq m_0$ then $S \geq m$, i.e., all problems of size (m, S) are terminal (do not involve further recursive calls to FAST-PEBBLE).

(b) If $m \geq m_0$ then $p \geq 1$, i.e., for all problems in which Case 2 applies, the pool P is not empty.

(c) If $m \geq m_0$ then for each recursive call to FAST-PEBBLE with problem size (m', S') we have $S' \geq c_5 \log d\,(m'/\log m')$, i.e. the condition on S given in the premise of this lemma is hereditary.

Proof: The proof is again technical.

(a) Because of constraint (A9) and $m \leq m_0$ we have

$$\frac{c_5 \log d}{\log m} \geq \frac{(1 + \log c_4) \log d}{\log c_4 + \log d} \geq 1.$$

(b) Because of constraint (A8) we have

$$\frac{c_5 \log d}{\log c_4 + \log d} \geq \sqrt{\frac{9}{2c_4(2c_2 - c_3)}}.$$

Using $m \geq m_0 \geq c_4 d$ we get

$$\frac{(2c_2 - c_3)(c_5 \log d)^2 m}{d\,(\log m)^2} \geq \frac{9}{2}.$$

Since $S \geq c_5 \log d\, m/\log m$ we have

$$\frac{(2c_2 - c_3) S^2}{dm} \geq \frac{9}{2}$$

and with $S \leq m$

$$p \geq \frac{(2c_2 - c_3) S^2}{dm} - \left(\frac{d+1}{d}\right)\left(\frac{S}{m}\right) - 2 \geq 1.$$

(c) First we prove that $m' \geq (\frac{1}{2} - c_1)m$. With $S \leq m$ and constraint (A7) we have for $m \geq m_0 \geq (d+1)/c_2$

$$\begin{aligned} m' &\geq m\left(\frac{1}{2} + \frac{S}{m}(c_2 - c_1) - \frac{d+1}{m}\right) \\ &\geq m\left(\frac{1}{2} + c_2 - c_1 - \frac{d+1}{m}\right) \\ &\geq \left(\frac{1}{2} - c_1\right)m. \end{aligned}$$

We know by Lemma 5.2.2 and part (b) of this lemma that

$$\frac{m'}{S'} \leq \frac{m}{S} - c_3.$$

Furthermore we have $S \geq c_5 \log d\,(m / \log m)$. By constraint (A6)

$$-c_3c_5 \log d \leq \log(\frac{1}{2} - c_1).$$

Thus

$$\log m' \geq \log(\frac{1}{2} - c_1) + \log m \geq \log m - c_3c_5 \log d.$$

It follows that

$$\frac{m'}{S'} \leq \frac{m}{S} - c_3 \leq \frac{\log m}{c_5 \log d} - c_3 \leq \frac{1}{c_5 \log d}(\log m - c_3c_5 \log d) \leq \frac{\log m'}{c_5 \log d}.$$

This proves (c). ∎

The upper bound on the time $T(G,S)$ it takes FAST-PEBBLE to pebble the graph G of size m with maximum in-degree d ($d \geq 2$) using S pebbles where ($S \geq c_5 \log d\, m/\log m$) can now be proved. Lemma 5.2.3 implies the correctness of FAST-PEBBLE whereas Lemma 5.2.2 gives the basis for an inductive argument for proving the time bound. The details are given in the following theorem.

Theorem 5.2.4: Let G and S be as defined above. FAST-PEBBLE pebbles G with S pebbles in time $T(G,S)$ where

$$T(G,S) \leq d^{2^{m/c_3S} - 1} \quad S \quad c_7^{c_8^{m/S}}. \tag{6}$$

Proof: The proof is by induction on the quantity

$$j := \left\lceil \left(\frac{m}{S} - 1 \right) \Big/ c_3 \right\rceil .$$

$j \leq 0$: In this case we have $m \leq S$, the problem is terminal and the theorem follows trivially.

$j > 0$: In this case we have $m > S$. FAST-PEBBLE calls itself recursively with arguments, say (m', S'). By Lemma 5.2.3 (a) we have $m \geq m_0$. By Lemma 5.2.3 (c) the problem size (m', S') also fulfills the premise of the theorem. By Lemma 5.2.3 (a),(b) and Lemma 5.2.2 we have

$$\frac{m'}{S'} \leq \frac{m}{S} - c_3 .$$

Thus

$$\begin{aligned} j' &:= \left\lceil \left(\frac{m'}{S'} - 1 \right) \Big/ c_3 \right\rceil \\ &\leq \left\lceil \left(\frac{m}{S} - c_3 - 1 \right) \Big/ c_3 \right\rceil \\ &= \left\lceil \left(\frac{m}{S} - 1 \right) \Big/ c_3 \right\rceil - 1 \\ &= j - 1. \end{aligned}$$

Thus the theorem can be applied inductively on all recursive calls. We have to make a case distinction corresponding to Case 1 (Small Cut) and Case 2 (Big Cut).

Case 1 (Small Cut): There is only one recursive call of FAST-PEBBLE on each of G_1 and G_2, both times with $\lceil (1 - c_1)S \rceil$ pebbles. Thus

$$\begin{aligned} T(G, S) &\leq T(G_1, \lceil (1 - c_1)S \rceil) + T(G_2, \lceil (1 - c_1)S \rceil) \\ &\leq 2\, d^{2^{m/(c_3 S)-1} - 1} \quad S \quad c_7 c_8^{m/S - c_3} \\ &\leq d^{2^{m/(c_3 S)-1}} \quad S \quad c_7 c_8^{m/S - c_3} \qquad (\text{since } d \geq 2) \\ &\leq d^{2^{m/c_3 S} - 1} \quad S \quad c_7 c_8^{m/S} \qquad (\text{since } m/c_3 S \geq 1). \end{aligned}$$

Case 2 (Big Cut): By Lemma 5.2.3 (b) we have $p \geq 1$. In fact by the definition of p

$$p = \left\lfloor \left(2c_2 - c_3 - \frac{d}{S}\right)\frac{S^2}{dm} \right\rfloor - 1$$

$$\geq \left(2c_2 - c_3 - \frac{d}{S}\right)\frac{S^2}{3dm}$$

since $\lfloor z \rfloor - 1 \geq z/3$ if $z \geq 2$. Since $S \geq c_5 \log d\, m / \log m$ and $m \geq m_0 \geq c_4 d$ we can bound d/S from above and get

$$p \geq c_8 \frac{S^2}{dm} \tag{7}$$

where $c_8 > 0$ is a constant that fulfills constraint (A10).

By the definition of FAST-PEBBLE, in the case of a big cut G_2 is pebbled once with $\lfloor \frac{S-dp}{2} \rfloor$ pebbles and G_1 is pebbled at most $\lceil T(G_2, \lfloor \frac{S-dp}{2} \rfloor)/p \rceil$ times, namely at most once for each p (consecutive) placements on G_2. Thus

$$T(G,S) \geq T\left(G_2, \left\lfloor \frac{S-dp}{2} \right\rfloor\right) + T\left(G_1, \left\lfloor \frac{S-dp}{2} \right\rfloor\right) \left\lceil T\left(G_2, \left\lfloor \frac{S-dp}{2} \right\rfloor\right) \Big/ p \right\rceil.$$

Applying the theorem inductively to the recursive calls and using Lemma 5.2.2 and the estimate (7) for p we get

$$T(G,S) \leq d^{2^{m/(c_3 S)-1} - 1} S\, c_7^{c_8^{m/S-c_3}}$$

$$+ d^{2^{m/(c_3 S)-1} - 1} S\, c_7^{c_8^{m/S-c_3}} \left\lceil \frac{dm}{c_8 S^2} \left(d^{2^{m/(c_3 S)-1} - 1} S\, c_7^{c_8^{m/S-c_3}} \right) \right\rceil$$

and eliminating the ceiling function, multiplying out and collecting terms

$$T(G,S) \leq 2\, d^{2^{m/(c_3 S)-1}} S\, c_7^{c_8^{m/S-c_3}} + \left(\frac{1}{c_8}\right) S\, d^{2^{m/c_3 S} - 1} \left(\frac{m}{S}\right) c_7^{2c_8^{m/S-c_3}}.$$

Since $m/S \geq 1$ and c_7 and c_8 fulfill constraint (A12) we get

$$T(G,S) \leq d^{2^{m/c_3 S} - 1}\ S\ \left(1 + \frac{1}{c_8}\right)\left(\frac{m}{S}\right)\ c_7^{2c_8^{m/S-c_3}}. \tag{8}$$

The inequality (8) differs from the desired formula (6) only by an additional factor of

$$\left(1 + \frac{1}{c_8}\right)\left(\frac{m}{S}\right)$$

on the base line, the factor of 2 in the exponent of c_7 and the subtraction of c_3 in the exponent of c_8.

The crucial step in the argument (and the step which requires T to grow as a double exponential in (m/S)) is to show that adding c_3 to the exponent of c_8 outweighs the cancellation of both the factor of $(1+\frac{1}{c_6})(\frac{m}{S})$ on the base line and the factor of 2 in the exponent of c_7. Formally this is stated in the following fact whose proof is again technical.

Fact 5.2.5: If $m/S \geq 1$ then

$$c_7^{(c_8^{c_3}-2)}\, c_8^{m/S-c_3} \geq \left(1+\frac{1}{c_6}\right)\left(\frac{m}{S}\right). \tag{9}$$

Proof of Fact 5.2.5: The expression

$$a^{b^x}\Big/ x \tag{10}$$

considered as a function of x has the following first derivative.

$$\frac{a^{b^x}}{x^2}\left((\ln a)\,(\ln b)\, x\, b^x - 1\right).$$

If this derivative is greater than 0 for all $x \geq 1$ then a minimum of the function (10) for $x \geq 1$ is given by its value at $x = 1$. If we substitute

$$a = c_7^{1-2/c_8^{c_3}} \qquad b = c_8 \qquad x = \frac{m}{S}$$

into (10) the resulting function is the function

$$\left(1+\frac{1}{c_6}\right)\frac{\text{left hand side of (9)}}{\text{right hand side of (9)}}. \tag{11}$$

Constraints (A12) (A13) and (A14) assert that its derivative is positive for all $m/S \geq 1$. Thus the minimum of (11) is given by its value at $m/S = 1$. By constraint (A11) this value is greater than $(1+\frac{1}{c_6})$, which proves (9). ∎

We now continue the proof of Theorem 5.2.4. Using Fact 5.2.5 it follows that

$$\begin{aligned} T(G,S) &\leq d^{2^{m/c_3 S}-1}\ S\ c_7^{c_8^{m/S-c_3}}\ \left(1+\frac{1}{c_6}\right)\left(\frac{m}{S}\right) \\ &\leq d^{2^{m/c_3 S}-1}\ S\ c_7^{2c_8^{m/S-c_3}}\ \left(c_7^{(c_8^{c_3}-2)}\, c_8^{m/S-c_3}\right) \\ &\leq d^{2^{m/c_3 S}-1}\ S\ c_7^{c_8^{m/S}}. \end{aligned}$$

This completes the proof of the theorem. ∎

It should be mentioned that at the expense of the simplicity of the argument the constants in the upper bound given in Theorem 5.2.4 can be improved. However, we are mainly interested in the asymptotic behaviour of the bound as described by the following corollary.

Corollary 5.2.6: If the premise of Theorem 5.2.4 is fulfilled then

$$T(G,S) \leq S\,(c_7 d)^{c_8^{(d+1)N/S}}$$

where N denotes the number of vertices in G.

Proof: Since G has a maximum in-degree d we have $m \leq (d+1)N$. Using Theorem 5.2.4 we get

$$\begin{aligned}
T(G,S) &\leq d^{2^{m/c_3 S}-1} \quad S \quad c_7^{c_8^{m/S}} \\
&\leq S\, d^{(2^{1/c_3})^{m/S}}\, c_7^{c_8^{m/S}} \\
&\leq S\, d^{c_8^{m/S}}\, c_7^{c_8^{m/S}} \qquad \text{by constraint (A13)} \\
&\leq S\,(c_7 d)^{c_8^{m/S}} \\
&\leq S\,(c_7 d)^{c_8^{(d+1)N/S}}. \qquad \blacksquare
\end{aligned}$$

For any constant $d \geq 2$, if $S \geq cN/\log\log N$ for a sufficiently large constant c (depending on d) then $T(G,S)$ is polynomial in N.

5.3 The Lower Bound In The Black & White Pebble Game

According to Section 5.1 the graph family $C(n,k)$ defined in Chapter 4 does not exhibit the worst time-space tradeoff possible. The graphs $C(n,k)$ can be pebbled with $S \geq cN \log\log N/\log N$ pebbles in polynomial time if $c > 0$ is large enough. On the other hand, if $S_J = \Omega(N/\log\log N)$ (as is stated in Section 5.1) then graph families have to exist whose pebbling time is superpolynomial whenever $S \leq cN/\log\log N$ for small enough $c > 0$. (Observe that $N \log\log N/\log N = o(N/\log\log N)$.) In order to prove the lower bounds stated in Section 5.1 we therefore have to find graph families which have even more dramatic time-space tradeoffs than $C(n,k)$.

The essential idea in the construction and analysis of $C(n,k)$ was to arrange as many superconcentrators as possible in levels one below the other, such that the BLBA can be iterated through the levels. Intuitively, the more levels the graph has, the more frequently the BLBA can be iterated and the better the lower bound on the pebbling time should be. In Chapter 4 we chose $k = \Theta(N/S)$ and $n = \Theta(S)$ as this maximized the time. However, since $S = \Omega(N/\log N)$ we always have $k = O(\log N)$; thus the worst graphs among the $C(n,k)$ do not have very many levels. It is suggested that we should find some way of increasing the number of levels without increasing the graph size. Obviously the only way in which this is possible is to use superconcentrators of different sizes. They have to be arranged and interconnected in such a fashion as to retain the ability to iterate the BLBA through practically all levels (or at least some constant fraction of them). This means that long edges have to be introduced that connect non-adjacent levels.

These ideas lead to the following definition of a suitable graph family $G(n,k)$ (again in two parameters), that is somewhat reminiscent of the graph family used in [PTC77] to prove the space lower bound.

As in Section 5.2 we have to use a number of constants $c_1, \ldots, c_{10} > 0$ satisfying certain constraints. A list of the constraints and short explanations of the significance of the constants are given in Appendix B. An example for a set of constants satisfying all constraints is:

$$c_1 = \frac{9}{11} \qquad c_2 = \frac{1}{11} \qquad c_3 = \frac{1}{44} \qquad c_4 = \frac{1}{88} \qquad c_5 = \frac{1}{180}$$

$$c_6 = \frac{1}{2} \qquad c_7 = 32500 \qquad c_8 = 1.03 \qquad c_9 = 1.00001 \qquad c_{10} = \frac{1}{750}$$

The graphs $G(n,k)$ are defined as follows.

Definition 5.3.1: Let n be divisible by $\lceil 1/c_2 \rceil 2^k$. $G(n,k)$ is inductively defined as follows:

(a) $G(n,1)$ is Pippenger's linear n-superconcentrator.

(b) For $k > 1$, $G(n,k)$ contains three copies C_{hi}, C_{med} and C_{lo} of Pippenger's linear n-superconcentrator and two copies G_{hi} and G_{lo} of $G(n/2, k-1)$. Let the inputs of C_i be denoted by $\sigma_{1,C_i}, \ldots, \sigma_{n,C_i}$ and its outputs by $\tau_{1,C_i}, \ldots, \tau_{n,C_i}$ $(i \in \{\text{hi}, \text{med}, \text{lo}\})$. Adopt corresponding notation for G_i $(i \in \{\text{hi}, \text{lo}\})$. Then

$G(n,k)$ contains the following additional edges:

$$\{(\tau_{i,C_{hi}}, \sigma_{i,C_{med}}), (\tau_{i,C_{med}}, \sigma_{i,C_{lo}}) \mid 1 \leq i \leq n\}$$
$$\cup \{(\tau_{i,C_{hi}}, \sigma_{i,G_{hi}}), (\tau_{i+n/2,C_{hi}}, \sigma_{i,G_{hi}}),$$
$$(\tau_{i,G_{hi}}, \sigma_{i,C_{med}}), (\tau_{i,G_{hi}}, \sigma_{i+n/2,C_{med}}),$$
$$(\tau_{i,C_{med}}, \sigma_{i,G_{lo}}), (\tau_{i+n/2,C_{med}}, \sigma_{i,G_{lo}}),$$
$$(\tau_{i,G_{lo}}, \sigma_{i,C_{lo}}), (\tau_{i,G_{lo}}, \sigma_{i+n/2,C_{lo}}), \mid 1 \leq i \leq n/2\}$$

An illustration of this construction is given in Figure 9.

It is easily proved by induction on k that $G(n,k)$ has between $(8k-4)n$ and $(120k-80)n$, i.e., $\Theta(nk)$ vertices and furthermore that $G(n,k)$ has $2^{k+1}-3$ superconcentrators.

If we unfold the recurrence in Definition 5.3.1 we see that can represent $G(n,k)$ schematically (leaving out all edges between superconcentrators) as a bar graph, where each bar represents a superconcentrator.

Definition 5.3.2: To each superconcentrator C in $G(n,k)$ we assign a *level number* $l_k(C)$ as follows:

(a) If $k = 1$ then $G(n,k)$ is a superconcentrator C and
$l_k(C) := 1$;

(b) If $k > 1$ then
$l_k(C_{hi}) := 1$;

for all superconcentrators C in G_{hi}
$l_k(C) := l_{k-1}(C) + 1$;

$l_k(C_{med}) := 2^k - 1$;

for all superconcentrators C in G_{lo}
$l_k(C) := l_{k-1}(C) + 2^k - 1$;

$l_k(C_{lo}) := 2^{k+1} - 3$.

In effect Definition 5.3.2 numbers the superconcentrators from top to bottom in the bar graph of $G(n,k)$. Figure 10 shows the bar graph of $G(n,k)$ for $k = 4$.

From now on we will denote the superconcentrator at level i by C_i. We will say that C_i *follows* C_j or that C_j *precedes* C_i if C_i is located below C_j in the bar graph of $G(n,k)$, i.e., if $i > j$.

Even though we were able to motivate the definition of $G(n,k)$ with observations about the graphs $C(n,k)$, no parts of the lower bound proof for $C(n,k)$ can be carried over to $G(n,k)$. We have to arrange the iterations of the BLBA in a different fashion. The following definitions introduce the essential concepts for the lower bound proof for $G(n,k)$.

Definition 5.3.3: Let C_i be a superconcentrator such that $i > 1$.

(a) The *parent* of C_i is the highest level superconcentrator preceding C_i that is larger than C_i (if such a superconcentrator exists, otherwise of the same size as C_i). The transitive closure of the parent relation is called the *ancestor* relation.

(b) The *neighborhood* of the superconcentrator C_i is the set of superconcentrators including C_i, its parent, and all superconcentrators preceding C_i and following its parent.

Definition 5.3.3 gives each level except the first one a parent and a neighborhood. The parent of C_i is larger than C_i, unless C_i is an n-superconcentrator, i.e., of the largest size possible in $G(n,k)$. (For example, in Figure 10, C_{17}, C_{19} and C_{21} have the same parent C_{18}. The neighborhoods of C_{17}, C_{19} and C_{21} are the sets $\{C_{16}, C_{17}\}$, $\{C_{16}, C_{17}, C_{18}, C_{19}\}$ and $\{C_{17}, C_{18}, C_{19}, C_{20}, C_{21}\}$ respectively. The parent of C_{29} is C_{15}.)

Again note, that we will count placements as well as removals of pebbles. (The same remarks as in Section 2.4 apply.) Assume that $S \leq c_3 n$. In the lower bound proof we will again analyze the distribution of pebbles on the graph.

Focussing our attention on level i of $G(n,k)$ we will consider time intervals during which many outputs of some superconcentrator C_j $(j < i)$ preceding C_i have to be pebbled while a lot of pebbles are bound on levels following i and thus not available for pebbling C_j. Again we will use the BLBA to insure that many outputs of C_j have to be pebbled during the time interval considered.

The appropriate definitions for carrying out the program outlined above are the following.

Definition 5.3.4: Let $Z = [z_1, z_2]$ be a time interval.

(a) An m-superconcentrator C_i is called *good* in Z if in each subinterval of Z in which at least $\lceil c_2 m \rceil$ outputs of C_i are pebbled, the number of pebbles on the neighborhood of C_i drops below $c_3 m$.

(b) An m-superconcentrator C_i is called *useful* in Z if C_i and all its ancestors are good in Z and at least $c_1 m$ outputs of C_i are pebbled in Z.

(c) Let C' be the parent of C_i. Assume that C' is not C_{i-1}. C' is *right* for C_i in Z if C' is useful in Z and at all times during Z at least $c_4 m$ pebbles stay on the neighborhood of C_i.

By constraints (B5) and (B6) $C_{2^{k+1}-3}$ is useful in the interval Z_{all} covering the whole pebbling strategy. We will show in the following that if C_i is useful in Z we can identify two disjoint subintervals of Z during each of which either C_{i-1} is useful or the parent of C_i is right for C_i. This allows us to iterate the argument through the levels from $C_{2^{k+1}-3}$ towards C_1 and at each step double the number of necessary placements. We have to pay for making a large jump (from C_i to its parent) by "losing" a proportional number of pebbles that are bound on the neighborhood of C_i during the interval considered. The number of available pebbles then implies an upper bound on the number of large jumps we can make, and thus gives a lower bound on the number of times the argument can be iterated. Each iteration doubles the number of necessary placements. The number of possible iterations will ultimately lead to the desired lower bound on the time-space tradeoff for pebbling $G(n,k)$.

In order to formally pursue this argument we have to prove a series of lemmas. The first lemma puts the BLBA into a context that is suitable for the discussion of $G(n,k)$.

Lemma 5.3.5: Let $i > 1$. Assume that in the interval Z, $\lceil c_2 m \rceil$ outputs of the m-superconcentrator C_i have to be pebbled starting and ending with a configuration of fewer than $2c_3 m$ pebbles on the neighborhood of C_i. Then during Z at least $c_1 m'$ outputs of the m'-superconcentrator that is the parent of C_i have to be pebbled.

Proof: By constraint (B3) we can apply the BLBA to Z and get that at least $(1 - 4c_3)m$ inputs of C_i have to be pebbled and unpebbled during Z. We have to make a case distinction.

Case 1: C_i is a C_{hi}. Thus C_{i-1} is the parent of C_i and because of the edges between C_{i-1} and C_i during Z at least $(1 - 5c_3)\,2m \geq 2c_1 m$ (see constraint (B4)) outputs of C_{i-1} have to be pebbled.

Case 2: C_i is a C_{med} resp. a C_{lo}. In this case let C' be the corresponding C_{hi} resp. C_{med} (i.e., $C' = C_{i-2^k m/n-2}$). Because of the direct connections between C' and C_i at least $(1 - 6c_3)m \geq c_1 m$ (see constraint (B4)) outputs of C' have to be

pebbled during Z. If C' is the parent of C_i then the lemma is proved. Otherwise, because of constraint (B2), we can apply the BLBA to Z again to show that at least $(1 - 4c_3)m$ inputs of C' are pebbled in Z. Repeated application of the case distinction to C' then proves the lemma. ∎

The following lemma gives the definition of usefulness its significance.

Lemma 5.3.6: Let $i > 1$. If the m-superconcentrator C_i is useful in Z then its parent is also useful in Z.

Proof: Since the parent of C_i is by definition good in $Z = [z_1, z_2]$ we only have to show that at least $c_1 m'$ outputs of the parent of C_i are pebbled in Z, where m' is the number of inputs of the parent of C_i.

Since $n/2^k \geq 1/c_2$ (see Definition 5.3.1) we have $c_2 m \geq c_2 n/2^{k-1} \geq 2$ and thus $6\lceil c_2 m \rceil \leq 9 c_2 m \leq c_1 m$ (using constraint (B2)), i.e., at least $6\lceil c_2 m \rceil$ outputs of C_i are pebbled in Z.

Let Z_1 be the interval during which the first $\lceil c_2 m \rceil$ outputs of C_i are pebbled. Let Z_2 be the interval during which the next $\lceil c_2 m \rceil$ outputs of C_i are pebbled, and let Z_3 be the rest of Z. Then at least $\lceil c_2 m \rceil$ outputs are pebbled during Z_3. Furthermore, because C_i is good during Z, sometime during Z_1 and sometime during Z_3 the number of pebbles on the neighborhood of C_i (i.e., certainly the number of pebbles on C_i itself) drops below $c_3 m$. Let this happen at time z_0 in Z_1 and at time z_3 in Z_3. During the time interval $[z_0 + 1, z_3] \subset Z$ at least $\lceil c_2 m \rceil$ outputs of C_i are pebbled starting with a configuration of fewer than $c_3 m$ pebbles on C_i. Applying Lemma 5.3.5 to the interval $[z_0 + 1, z_3]$ yields the result. ∎

We will now prove the lemma that provides us with a lower bound argument that can be iterated on $G(n, k)$.

Lemma 5.3.7: Let $i > 1$. Let the m-superconcentrator C_i be useful in Z. Then there are two disjoint subintervals Z'_1 and Z'_2 of Z such that in Z'_j $(j \in \{1, 2\})$ either C_{i-1} is useful or the parent of C_i is right for C_i.

Proof: If C_{i-1} is the parent of C_i then the lemma follows from Lemma 5.3.6. Let us thus assume that this is not the case.

Then C_{i-1} has half the size of C_i. Because C_i is useful in Z, as in the proof of Lemma 5.3.5, at least $6\lceil c_2 m \rceil$ outputs of C_i are pebbled in Z. Let $Z_1 := [z_{1,1}, z_{2,1}]$ be the interval in which the first $3\lceil c_2 m \rceil$ outputs of C_i are pebbled and let $Z_2 := [z_{1,2}, z_{2,2}]$ be the interval in which the last $3\lceil c_2 m \rceil$ outputs of C_i are pebbled. The following argument can be applied equally to both Z_j $(j \in \{1, 2\})$ and we will without loss of generality consider only Z_1.

Let z_0 be the first time and z_1 be the last time in Z_1 at which fewer than c_3m pebbles are on the neighborhood of C_i. Because C_i is good in Z at least $\lceil c_2m \rceil$ outputs of C_i are pebbled in $Z' = [z_0 + 1, z_1]$. We make the following case distinction.

Case 1 (C_{i-1} is bad in Z'): Suppose that there is a subinterval $[z_2, z_3] \subset Z'$ during which $\lceil c_2m/2 \rceil$ outputs of C_{i-1} are pebbled and always at least c_4m pebbles stay on the neighborhood of C_i. Without loss of generality we can assume that at time z_2 a pebble is placed on the neighborhood of C_i. Let z_4 be the last time before (and not including) z_2 at which there are fewer than c_3m pebbles on the neighborhood of C_i. Let z_5 be the first time after (and including) z_3 at which there are fewer than c_3m pebbles on the neighborhood of C_i. Let $Z'' := [z_4 + 1, z_5]$ and let $Z_1' := [z_4 + 1, z_6]$ where $z_6 = z_5$ if $z_5 = z_3$ and $z_6 = z_5 - 1$ otherwise. We have $Z_1' \subset Z'' \subset Z'$. (Furthermore note that if $z_6 = z_5 - 1$ then at z_5 a pebble is removed from the neighborhood of C_i.) During Z'', $\lceil c_2m/2 \rceil$ outputs of C_{i-1} are pebbled starting and ending with configurations of fewer than c_3m pebbles on the neighborhood of C_i (i.e., also on the neighborhood of C_{i-1}). Applying Lemma 5.3.5 to Z'' yields that at least c_1m outputs of the parent of C_{i-1} are pebbled during Z''. Applying Lemma 5.3.5 again (using constraint (B2)) yields that at least $2c_1m$ outputs of the grandparent of C_{i-1}, i.e., the parent of C_i are pebbled in Z'', i.e., also in Z_1'. Furthermore at all times during Z_1' at least c_4m pebbles stay on the neighborhood of C_i. Thus the parent of C_i is right for C_i in Z_1'.

Case 2 (C_{i-1} is good in Z'): Otherwise any time in Z' that $\lceil c_2m/2 \rceil$ outputs of C_{i-1} are pebbled the number of pebbles on the neighborhood drops below c_4m. Because of constraint (B5) this means that C_{i-1} is good in Z'. We make another case distinction.

Case 2.1 (C_i is large): The parent of C_i has the same size as C_i. In this case the parent of C_i is also the parent of C_{i-1}. Thus C_{i-1} and all its ancestors are good in Z'. But in Z' we have to pebble $\lceil c_2m \rceil$ outputs of C_i starting and ending with configurations of fewer than c_3m pebbles on the neighborhood of C_i. By the BLBA we have to pebble in Z' at least $(1 - 2c_3)m$ inputs of C_i, and because of the edges between C_{i-1} and C_i at least $(1 - 4c_3)m/2$ outputs of C_{i-1}. Because of constraint (B4), C_{i-1} is useful in $Z_1' := Z'$.

Case 2.2 (C_i is small): The parent of C_i is larger than C_i. We have to make a third case distinction.

Case 2.2.1 (The parent of C_i is bad in Z'): Assume that there is a subinterval $[z_7, z_8] \subset Z'$ in which $\lceil c_2m \rceil$ outputs of the parent of C_{i-1} are pebbled while always

at least c_4m pebbles stay on the neighborhood of C_i. Without loss of generality we can assume that at time z_7 a pebble is placed on the neighborhood of C_i. Let z_9 be the last time before (and not including) z_7 at which there are fewer than c_3m pebbles on the neighborhood of C_i. Let z_{10} be the first time after (and including) z_8 at which there are fewer than c_3m pebbles on the neighborhood of C_i. Let $Z'' := [z_9 + 1, z_{10}]$ and $Z'_1 := [z_9 + 1, z_{11}]$ where $z_{11} = z_{10}$ if $z_{10} = z_8$ and $z_{11} = z_{10} - 1$ otherwise. We have $Z'_1 \subset Z'' \subset Z'$. (Furthermore note that if $z_{11} = z_{10} - 1$ then at z_{10} a pebble is removed from the neighborhood of C_i.) During Z'', $\lceil c_2m \rceil$ outputs of the parent of C_{i-1} are pebbled starting and ending with a configuration of fewer than c_3m pebbles on the neighborhood of C_i (i.e., also on the neighborhood of C_{i-1}). Applying Lemma 5.3.5 to Z'' yields that at least $2c_1m$ outputs of the grandparent of C_{i-1} (i.e., the parent of C_i) are pebbled during Z'', i.e., also during Z'_1. Furthermore at all times during Z'_1 at least c_4m pebbles stay on the neighborhood of C_i. Thus the parent of C_i is right for C_i in Z'_1.

Case 2.2.2 (The parent of C_i is good in Z'): Otherwise any time in Z' that $\lceil c_2m \rceil$ outputs of the parent of C_{i-1} are pebbled the number of pebbles on the neighborhood of C_i drops below c_4m. Thus the parent of C_{i-1} is good in Z', and thus C_{i-1} and all its ancestors are good in Z'. As in Case 2.1 we can infer that C_{i-1} is useful in $Z'_1 := Z'$. ∎

We are now able associate with the strategy for pebbling $G(n,k)$ a rooted binary tree according to the following definition.

Definition 5.3.8: Let R be the rooted binary tree such that each vertex in R is an ordered pair (i, Z) where $1 \leq i \leq 2^{k+1} - 3$, $Z \subset Z_{all}$ is a time interval and C_i is useful in Z. Furthermore

(a) The root of R is the vertex $(2^{k+1} - 3, Z_{all})$.

(b) Each vertex $v = (i, Z)$ in R such that $i > 1$ has two children (j, Z'_j), $j \in \{1, 2\}$ where Z_j is defined as in Lemma 5.3.7 and $C_{i_j} = C_{i-1}$ if C_{i-1} is useful in Z'_j, otherwise C_{i_j} is the parent of C_i (and C_{i_j} is right for C_i in Z'_j by Lemma 5.3.7).

(c) Each vertex $v = (1, Z)$ in R is a leaf.

Thus all leaves of R are vertices $v_i = (1, Z_i)$ where the Z_i are pairwise disjoint subintervals of Z_{all} during which C_1 is useful. By constraint (B2) at least $6\lceil c_2n \rceil$ outputs of C_1 are pebbled in each of the Z_i. The BLBA can be applied twice (see

of the Z_i. Thus if b is a lower bound on the number of leaves in R then $2(1-c_3)nb$ is a lower bound on the number of placements of pebbles on inputs of C_1, and therefore also a lower bound on the time necessary to pebble $G(n,k)$.

We can therefore prove the following theorem.

Theorem 5.3.9: In order to pebble $G(n,k)$ with $S \leq c_5 n$ pebbles, a time T is necessary such that

$$T \geq \left(\frac{1-c_3}{4}\right) \; n \; 2^{(1-2c_5/c_4)2^{k+1}}.$$

Proof: We will prove in the following lemma that R has at least $2^{(1-2c_5/c_4)2^{k+1}-3}$ leaves. ∎

Lemma 5.3.10: The number of leaves in R is bounded from below by

$$2^{(1-2c_5/c_4)2^{k+1}}.$$

Proof: Each non-leaf in R has two children. Thus if we prove that each path in R has a length of at least b than we can infer that there are at least 2^b leaves in R. Therefore the lemma follows from the following lemma. ∎

Lemma 5.3.11: Each path in R has a length of at least

$$\left(1-\frac{2c_5}{c_4}\right)2^{k+1}.$$

Proof: Consider an arbitrary path $p = (i_1, Z_1), \ldots, (i_r, Z_r)$ in R, where $i_1 = 2^{k+1} - 3$, $Z_1 = Z_{\text{all}}$, $i_r = 1$ and $Z_1 \supset Z_2 \supset \cdots \supset Z_r$. Let $I := \{i_1, \ldots, i_r\}$ and $B := \{1, \ldots, 2^{k+1} - 3\} - I$. Each level $v \in B$ fulfills $i_l > v > i_{l+1}$ for some l $(1 \leq l < r)$. Furthermore $C_{i_{l+1}}$ is the parent of C_{i_l} because by the definition of R otherwise $C_{i_{l+1}} = C_{i_l - 1}$, which is impossible. Thus $C_{i_{l+1}}$ is right for C_{i_l} in Z_l.

By Definition 5.3.4(c) if C_{i_l} is an $(n - 2^{k-j})$-superconcentrator then its neighborhood contains at least $c_4 n / 2^{k-j}$ pebbles during Z_l. But it also has at most $2^{j+1} - 2$ levels. All of these levels except i_l and i_{l+1} are in B.

Thus the operation of including a certain number of levels in B is accompanied by "using up" a certain number of pebbles that are permanently bound on those levels during $Z_r \subset Z_t$. We will express this fact by "charging" a proportional amount (namely $c_4 n/2^{k+1}$) of all pebbles bound on the neighborhood of C_{i_t} to each level in B that is in the neighborhood of C_{i_t}. This procedure can charge each pebble used for pebbling $G(n,k)$ to at most 2 levels (namely levels v and v' such that $i_{t-1} > v > i_t > v' > i_{t+1}$, if the pebble stays on C_{i_t} during Z_r). Therefore at least

$$|B|\, c_4 \frac{n}{2^{k+2}}$$

pebbles have to stay on $G(n,k)$ during Z_r. Since only $S \leq c_3 n$ pebbles are available, we get

$$|B| \leq \frac{c_5}{c_4} 2^{k+2}$$

and thus

$$r = 2^{k+1} - 3 - |B| \geq \left(1 - 2\frac{c_4}{c_5}\right) 2^{k+1} - 3. \qquad \blacksquare$$

We will now adjust k and n such that we can infer a lower bound from Theorem 5.3.9 that asymptotically matches the upper bound given in Section 5.2.

Let S be a function of N and let $N/\log N \leq S \leq N$. Choose $k = \lceil c_8 N/S \rceil$ where c_8 fulfills constraint (B7), and choose n to be the unique number in the interval $[\lceil 1/c_5 \rceil S, \lceil 1/c_5 \rceil S + \lceil 1/c_2 \rceil 2^k - 1]$ that is divisible by $\lceil 1/c_2 \rceil 2^k$. Define the graph $G'(N,S) := G(n,k)$. Then $G'(N,S)$ has at most

$$\begin{aligned}
120nk \leq & 120\left(c_8 \frac{N}{S} + 1\right)\left(\left\lceil \frac{1}{c_5} \right\rceil S + \left\lceil \frac{1}{c_2} \right\rceil 2^{c_8 N/S+1}\right) \\
\leq & 120\left(\left\lceil \frac{1}{c_5} \right\rceil c_8 N + c_8 \left\lceil \frac{1}{c_2} \right\rceil \frac{N}{S} 2^{c_8 N/S+1} + \left\lceil \frac{1}{c_5} \right\rceil S + \left\lceil \frac{1}{c_2} \right\rceil 2^{c_8 N/S+1}\right) \\
\leq & 120N\left(\left\lceil \frac{1}{c_5} \right\rceil (c_8 + 1) + 2c_8 \left\lceil \frac{1}{c_2} \right\rceil \log N\, N^{c_8 - 1} + 2N^{c_8 - 1}\right)
\end{aligned}$$

vertices. Maximization with respect to N yields that $G'(N,S)$ contains at most $c_7 N$ vertices, where c_7 fulfills constraint (B8).

Theorem 5.3.12: In order to pebble $G'(N,S)$ with S pebbles, where $N/\log N \leq S \leq N$ a time T is necessary such that

$$T \geq c_{10}\, S\, c_8^{c_9 M/S}$$

where M is the size of $G'(N,S)$, c_8 fulfills constraint (B9), c_9 fulfills constraint (B10) and c_{10} fulfills constraint (B11).

Proof: Substitute $n \geq \lceil 1/c_5 \rceil S$ and $k \geq c_9 N/S$ in Theorem 5.3.9. ∎

An analogous remark as in Section 4.2 applies here: $G'(N,S)$ has to be chosen in dependence of S. There is no single graph $G(n,k)$ that has a dramatic time space tradeoff for all S. The lower bound curve given in Theorem 5.3.12 is an envelope of the lower bound curves for the $G'(N,S)$ for all S. We strongly conjecture that there is no *single* graph whose time-space tradeoff has a lower bound as given in Theorem 5.3.12 *for all* S.

6 CONCLUSIONS

The results proved in Chapter 5 answer the basic open question in the area of graph pebbling—the question of the location of the range S_f where space savings become infeasible because the accompanying sacrifice in pebbling time is superpolynomial. In a certain sense this closes the area of general research concerned with graph pebbling. However, there are a number of generalizations and extensions that may be worth studying.

The upper and lower bounds proved in Chapter 5 are asymptotically tight, but the constants are very far apart. The two constants in the doubly exponential time bound are 29 and 40 in the upper bound and 1.03 and 1.00001 in the lower bound. With these constants the bounds are only of theoretical interest and become insignificant for practical purposes. However, we assume that especially the constants in the upper time bound can be improved significantly. The algorithm **FAST-PEBBLE** should in practice perform much better than the constants given in Section 5.2 suggest.

Furthermore, so far the graph families that realize superpolynomial lower time bounds—and in fact even the graph families discussed in [PTC77] that realize the lower space bound—are very hard to construct and rather obscure. Their essential element is the superconcentrator, and until very recently superconcentrators could not even be constructed at all. Their role in the context of practical algorithms is not very well understood. On the other hand, if we confine ourselves to the study of graph families that arise in algorithmic context and are simpler to construct, we can only derive much less dramatic time-space tradeoffs. It would be interesting to find out if the lower bounds given in Chapters 4 and 5 and in [PTC77] can also be realized by such graph families. There are two ways of trying to answer this questions affirmatively. Firstly, one could try to attach strong algorithmic significance to certain linear superconcentrators. Secondly, one could try to prove the lower bounds using other simpler graph families. On the other hand one could give evidence for the fact that practical graph families are indeed much easier to pebble than Chapters 4 and 5 and [PTC77] suggest, by defining a subset of "practical graphs" and proving better upper bounds for all graphs in this subset.

Part of the motivation for studying graph pebbling arose from connections between the pebble game and Turing-machine complexity (see [HPV77] and [Pa76]). In this research area upper bounds on Turing-machine time complexity are derived in two steps. In the first step certain Turing-machine computations are represented by directed acyclic graphs that are called *computation graphs*. In the second step a

Turing-machine is constructed that simulates a pebble game on these computation graphs with a certain given number of pebbles. We conjecture that the lower bounds in Chapter 5 and in [PTC77] can be realized with computation graphs. This would give some support towards conjecturing that the upper bounds on Turing-machine complexity that are derived in [HPV77] are tight.

The black & white pebble game is largely unexplored and several open questions have been posed by [CS76] and [Me78]. The most interesting question here is whether the addition of white pebbles can save more than a constant fraction of space on certain graphs (see Introduction). Also it can be noted that, while Chapters 4 and 5 and [GT78] show that white pebbles do not improve the general lower bounds, as long as one is dealing with specific graph families, white pebbles often seem to improve the time-space tradeoffs. Ladder graphs for instance can be pebbled with one black and one white pebble in linear time. And Chapter 2 exhibits an improvement also for bit reversal graphs if black and white pebbles are used.

Finally one can make investigations in graph pebbling more applicable by introducing slight modifications of the pebble game that implement certain specific features of machine architecture (like two-address instructions).

APPENDIX A

Table of constraints for the constants $c_1, \ldots, c_8 > 0$ used in Section 5.2:

$$c_3 < 2c_2 \tag{A1}$$

$$c_1 > 2c_2 \tag{A2}$$

$$c_1 + c_2 < \frac{1}{2} \tag{A3}$$

$$c_3 < \frac{1 - 2c_1 - 2c_2}{2 - 2c_1} \tag{A4}$$

$$c_4 \geq \frac{2}{(1 - 2c_1) - 2c_3(1 - c_1) - 2c_2} \tag{A5}$$

$$c_5 \geq -\frac{\log\left(\frac{1}{2} - c_1\right)}{c_3} \tag{A6}$$

$$c_4 \geq \frac{3}{2c_2} \tag{A7}$$

$$c_5 \geq (1 + \log c_4)\sqrt{\frac{9}{2c_4(2c_2 - c_3)}} \tag{A8}$$

$$c_5 \geq 1 + \log c_4 \tag{A9}$$

$$0 < 3c_6 \leq (2c_2 - c_3) - \frac{1 + \log c_4}{c_4 c_5} \tag{A10}$$

$$c_7(c_8^{c_3} - 2)c_8^{1-c_3} \geq 1 + \frac{1}{c_6} \tag{A11}$$

$$c_7 > 1, \qquad c_8 > 1 \tag{A12}$$

$$c_8^{2c_3} > 2 \tag{A13}$$

$$\left(1 - \frac{2}{c_8^{c_3}}\right) c_8 \ln c_7 \ln c_8 > 1 \tag{A14}$$

Explanation of the constants:

- c_1 constant factor in the critical size of the cutset E
- c_2 constant factor in the displacement of cut from the middle of G
- c_3 minimal decrease of m/S per recursion level
- c_4 constant factor in the threshold on the graph size for which **FAST-PEBBLE** becomes non-trivial
- c_5 constant factor in the lower bound on the size of S
- c_6 constant factor in the lower bound on the size of P
- c_7 first exponential in upper time bound
- c_8 second exponential in upper time bound.

APPENDIX B

Table of constraints for the constants $c_1, \ldots, c_{10} > 0$ used in Section 5.3:

$$c_1 < 1 \tag{B1}$$

$$9c_2 \leq c_1 \tag{B2}$$

$$c_3 \leq c_2/4 \tag{B3}$$

$$1 - 6c_3 \geq c_1 \tag{B4}$$

$$c_4 \leq c_3/2 \tag{B5}$$

$$c_5 < c_4/2 \tag{B6}$$

$$c_6 < 1 \tag{B7}$$

$$c_7 \geq 120\left(\left\lceil \frac{1}{c_5} \right\rceil (c_6 + 1) + \frac{2c_6}{1 - c_6} \exp\left(\frac{1 - c_6}{c_6 \lceil 1/c_2 \rceil} - 1\right)\right) \tag{B8}$$

$$c_8 \leq 2^{2 - 4c_3/c_4} \tag{B9}$$

$$c_9 \leq 2^{c_6/c_7} \tag{B10}$$

$$c_{10} \leq \frac{1 - c_3}{4\lceil 1/c_5 \rceil} \tag{B11}$$

Explanation of the constants:

- c_1 constant factor in the upper bound on the outputs pebbled (definition of usefulness)
- c_2 constant factor in the upper bound on the separation of two locally sparse configurations (definition of goodness)
- c_3 constant factor in the upper bound on the number of pebbles in locally sparse configurations (definiton of goodness)
- c_4 constant factor in the lower bound on the number of pebbles in locally dense configurations (definition of rightness)
- c_5 constant factor in the upper bound on S
- c_6 proportion for $k = \Theta(N/S)$
- c_7 constant factor in the upper bound to the size of $G'(N/S)$
- c_8 first exponential in the lower time bound
- c_9 second exponential in the lower time bound
- c_{10} constant factor in the lower time bound.

REFERENCES:

[AHU 75] A.V. Aho, J.E. Hopcroft, J.D. Ullmann, *The Design and Analysis of Computer Algorithms*, Addison-Wesley Publishing Company, Reading, Mass. (1975).

[Cha 73] A.K. Chandra, "Efficient compilation of linear recursive programs," *14th Annual Symposium on Switching and Automata Theory* (1973).

[Chu 78] F.R.K. Chung, "On concentrators, superconcentrators, generalizers, and non-blocking networks," Bell Laboratories, Murray Hill, N. J. (1978).

[Co 73] S.A. Cook, "An observation of time-storage tradeoff," *5th ACM-STOC* (1973), 29–33.

[CS 76] S.A. Cook, R. Sethi, "Storage requirements for deterministic polynomial finite recognizable languages," *JCSS* 13 (1976), 25–37.

[EGS 75] P. Erdös, R.L. Graham, E. Szemerédi, "On sparse graphs with dense long paths," *Comp. & Maths. with Appls.* 1 (1975), 365–369.

[EL 78] P. van Emde-Boas, J. van Leeuwen, "Move rules and trade-offs in the pebble game," Technical Report RW-CS-78-4, University of Utrecht, Utrecht, Netherlands (1978).

[GG 79] O. Gabber, Z. Galil, "Explicit construction of linear size concentrators and superconcentrators," personal communication (May 1979).

[GLT 79] J.R. Gilbert, T. Lengauer, R.E.Tarjan, "The pebbling problem is complete in polynomial space," *11th ACM-STOC* (1979), 237–248.

[GT 78] J.R. Gilbert, R.E. Tarjan, "Variations of a pebble game on graphs," Stanford Computer Science Report No. 661, Stanford University, Stanford, CA (1978).

[HP 70] C.E. Hewitt, M.S. Paterson, "Comparative schematology," *Project MAC Conf. on Concurrent Systems and Parallel Computation*, Woods Hole, Mass. (1970), 119–127.

[HPV 77] J.E.Hopcroft, W.J.Paul, L.G.Valiant, "On time versus space," *Journal ACM* 2 (1977), 332–337.

[LT 79] T. Lengauer, R.E. Tarjan, "Upper and lower bounds on time-space tradeoffs," *11th ACM-STOC* (1979), 262–277.

[Li 78] A. Lingas, "A P-space complete problem related to a pebble game," in *Automata, Languages and Programming* (Fifth colloquium, Undine 1978), Springer Lecture Notes in Computer Science No. 62 (1978) 300–321.

[Ma 73] G.A. Margulis, "Explicit constructions of concentrators," *Problemy Peredachi Informatsii* 9 (1973) 71–80 (English translation in: *Problems of Information Transmission*, Plenum. New York, 1975).

[Me 78] F. Meyer auf der Heide, "A comparison between two variations of a pebble game on graphs," University of Bielefeld, Bielefeld, West-Germany (1978).

[Pa 76] W.J. Paul, "On time hierarchies," *8th ACM-STOC* (1976), 218–222.

[Pin 73] M.S. Pinsker, "On the complexity of a concentrator," *Proc. Seventh International Teletraffic Congress*, Stockholm (1973), 318/1–318/4 (available from Secretariat, Televerket S-12386, Farsta, Sweden).

[Pip 77] N. Pippenger, "Superconcentrators," *SIAM J. Comp.* 6 (1977), 298–304.

[Pip 78] N. Pippenger, "A time-space tradeoff," *Journal ACM* 25 (1978), 509–515.

[PT 77] W.J.Paul, R.E. Tarjan, "Time-space tradeoffs in a pebble game," *Acta Informatica* 10 (1978), 111–115.

[PTC 77] W.J. Paul, R.E. Tarjan, J.R. Celoni, "Space bounds for a game on graphs," *Math. Sys. Th.* 10 (1977), 239–251.

[PV 76] N. Pippenger, L.G. Valiant, "Shifting graphs and their applications," *Journal ACM* 23 (1976), 423–432.

[Rei 78] R. Reischuk, "Improved bounds on the problem of time-space tradeoff in a pebble game," *19th IEEE-FOCS* (1978), 83–93.

[Se 75] R. Sethi, "Complete register allocation problems," *SIAM J. Comp.* 4 (1975), 226–248.

[SS 77] J.E. Savage, S. Swamy, "Space-time tradeoffs on the FFT-algorithm," Technical Report CS-31, Brown University, Providence, R.I (1977).

[SS 78a] J.E. Savage, S. Swamy, "Space-time tradeoffs for oblivious sorting and integer multiplication," Technical Report CS-32, Brown University, Providence, R.I. (1978).

[SS 78b] S. Swamy, J.E. Savage, "Space-time tradeoffs for linear recursion," Technical Report CS-36, Brown University, Providence, R.I. (1978).

[To 78] M. Tompa, "Time-space tradeoffs for computing functions, using connectivity properties of their circuits," *10th ACM-STOC* (1978), 196–204.

[Va 76] L.G. Valiant, "Graph-theoretic properties in computational complexity," *JCSS* 13 (1976), 278–285.

[Va 77] L.G. Valiant, "Graph-theoretic arguments in low level complexity," *Symposium on Math. Foundations of Computer Science*, Springer Lecture Notes in Computer Science No. 53, New York, N.Y. (1977), 162–176.

FIGURES

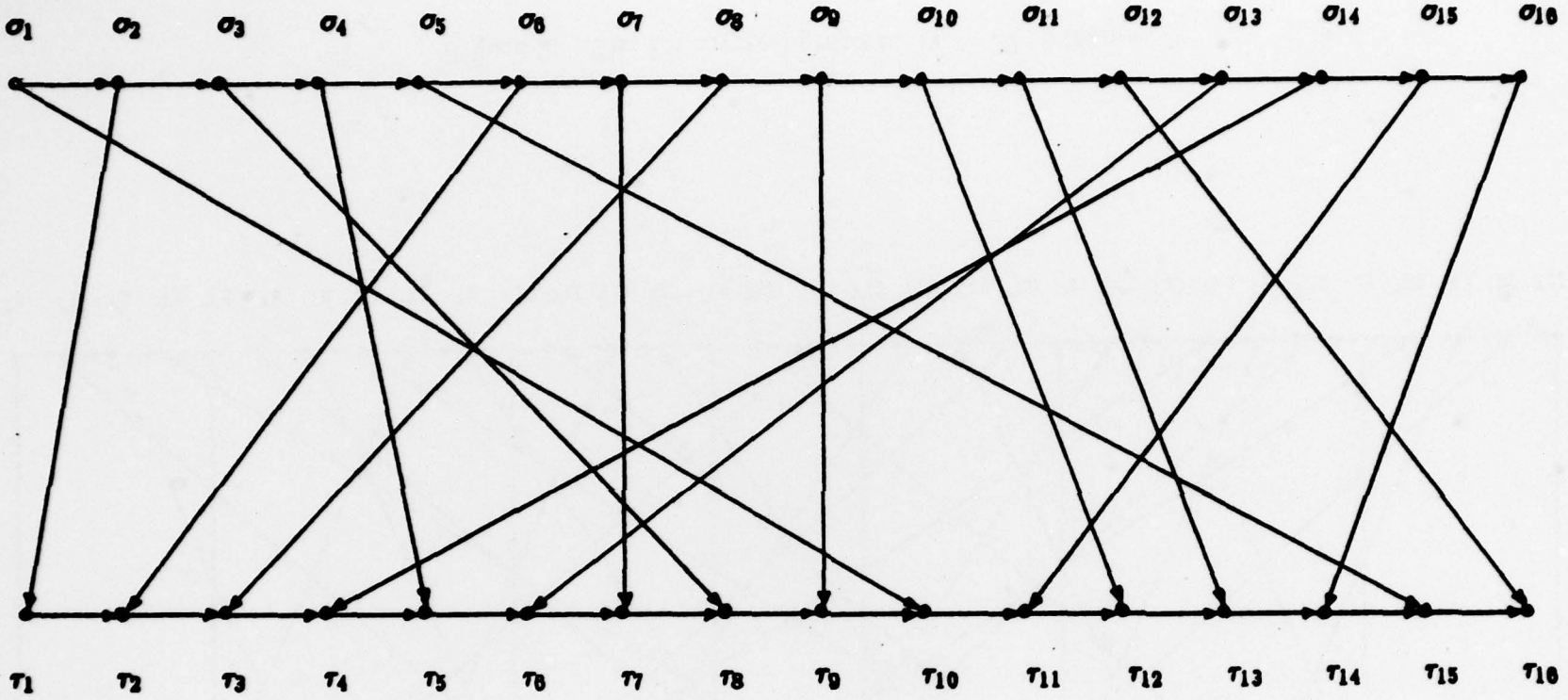


Figure 1: A typical permutation graph

$$N = 16, \qquad \pi = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16 \\ 10 & 1 & 8 & 5 & 15 & 2 & 7 & 3 & 9 & 10 & 13 & 16 & 6 & 4 & 11 & 14 \end{pmatrix}$$

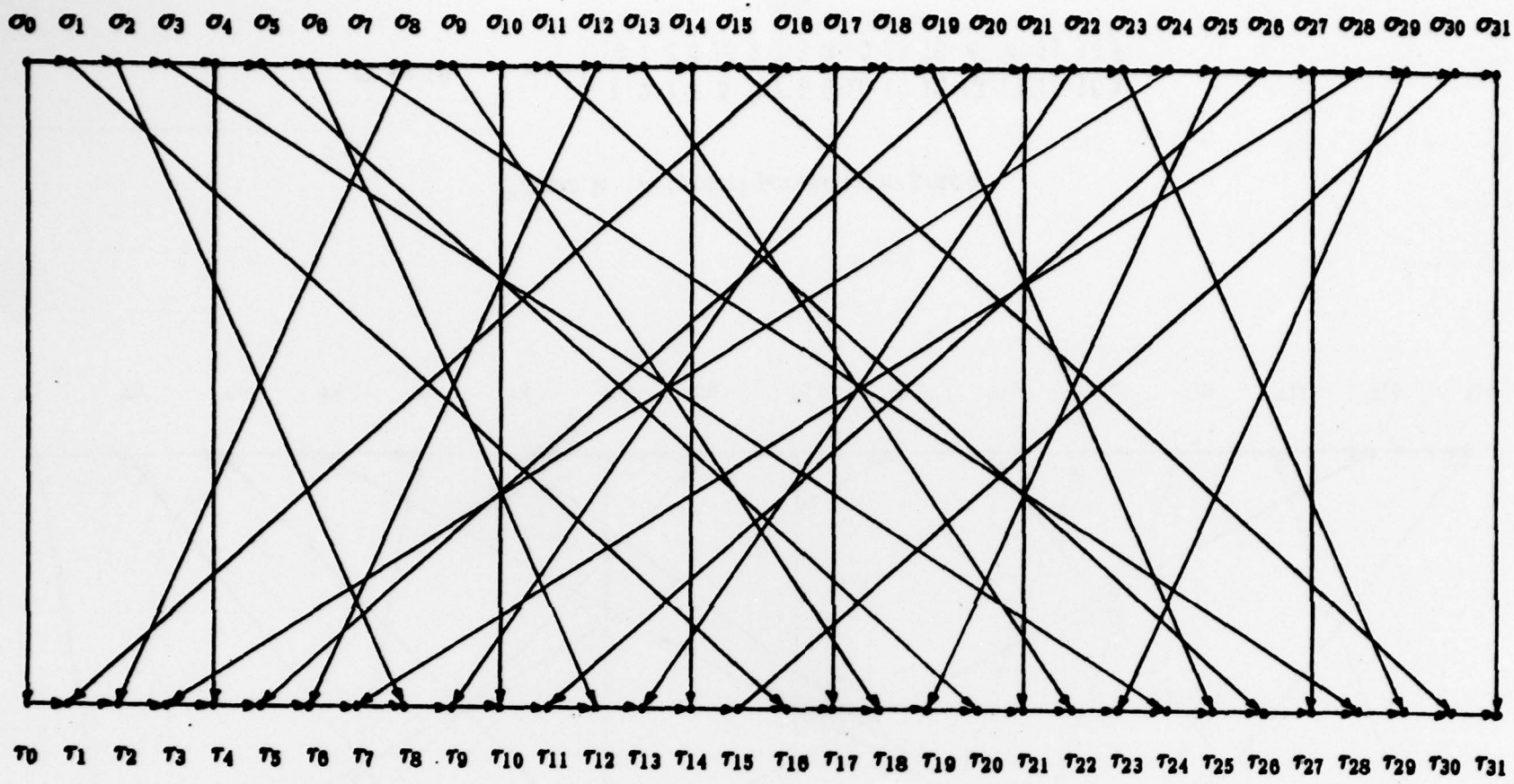


Figure 2: The bit reversal graph on $N = 32$ elements

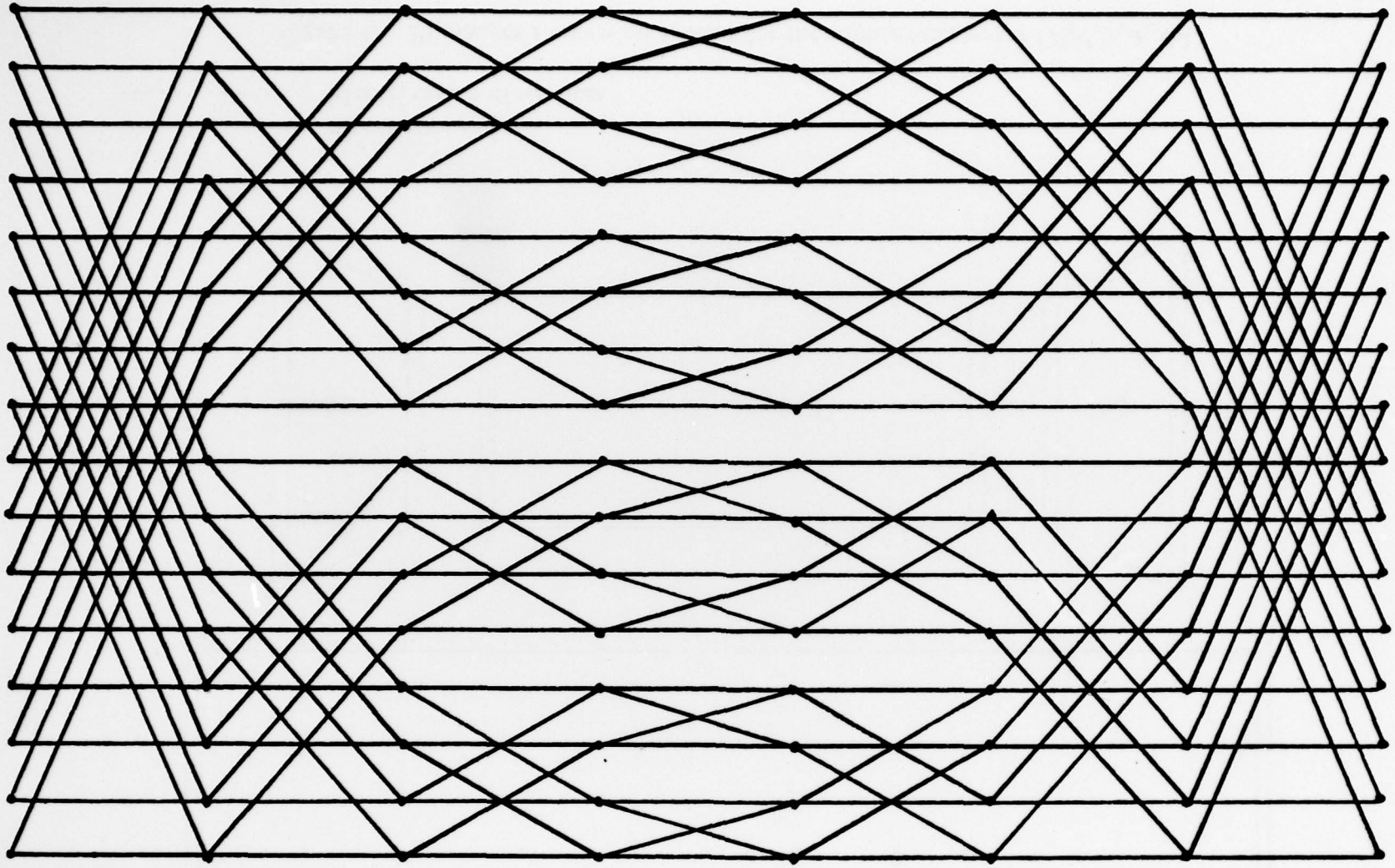
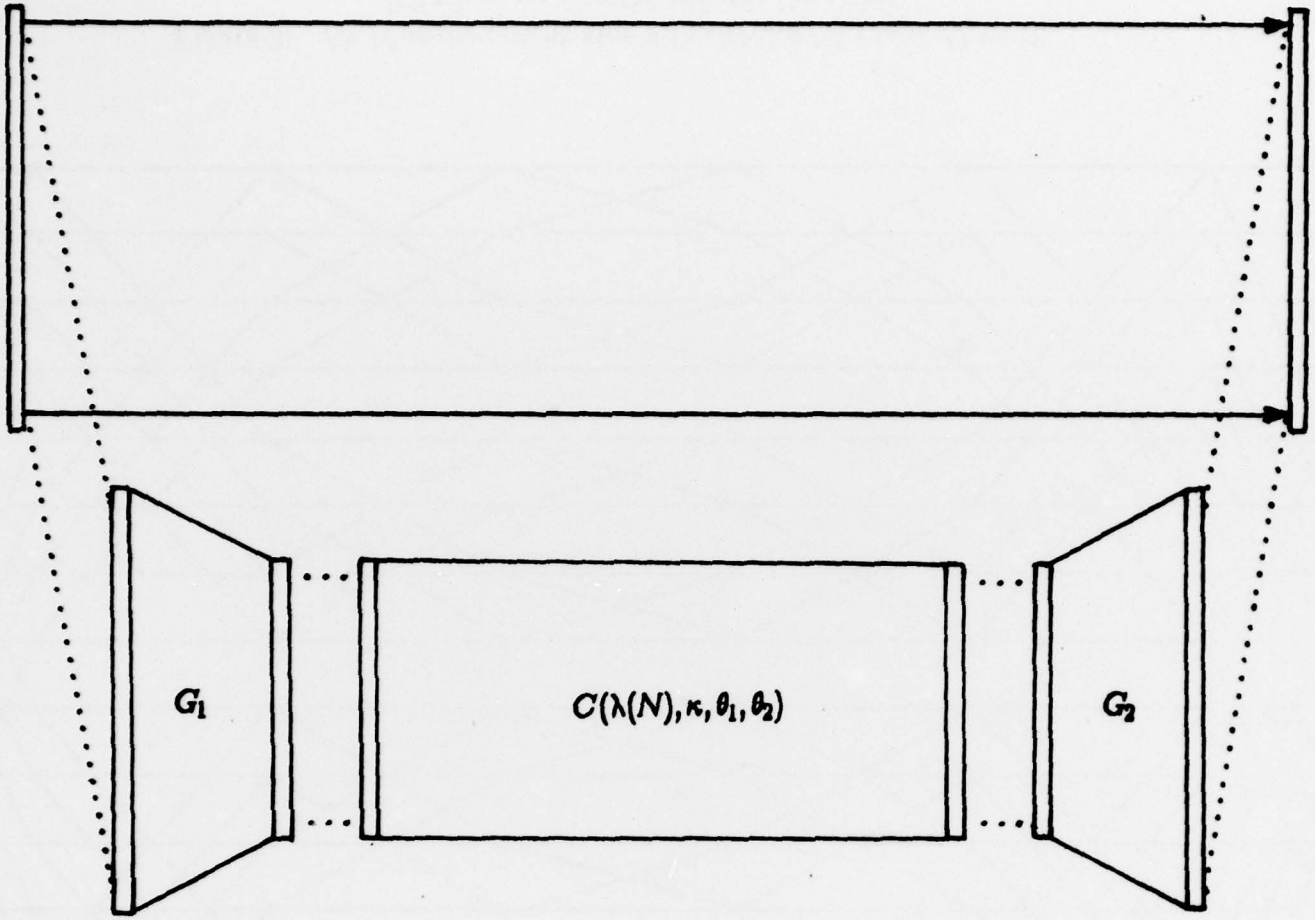


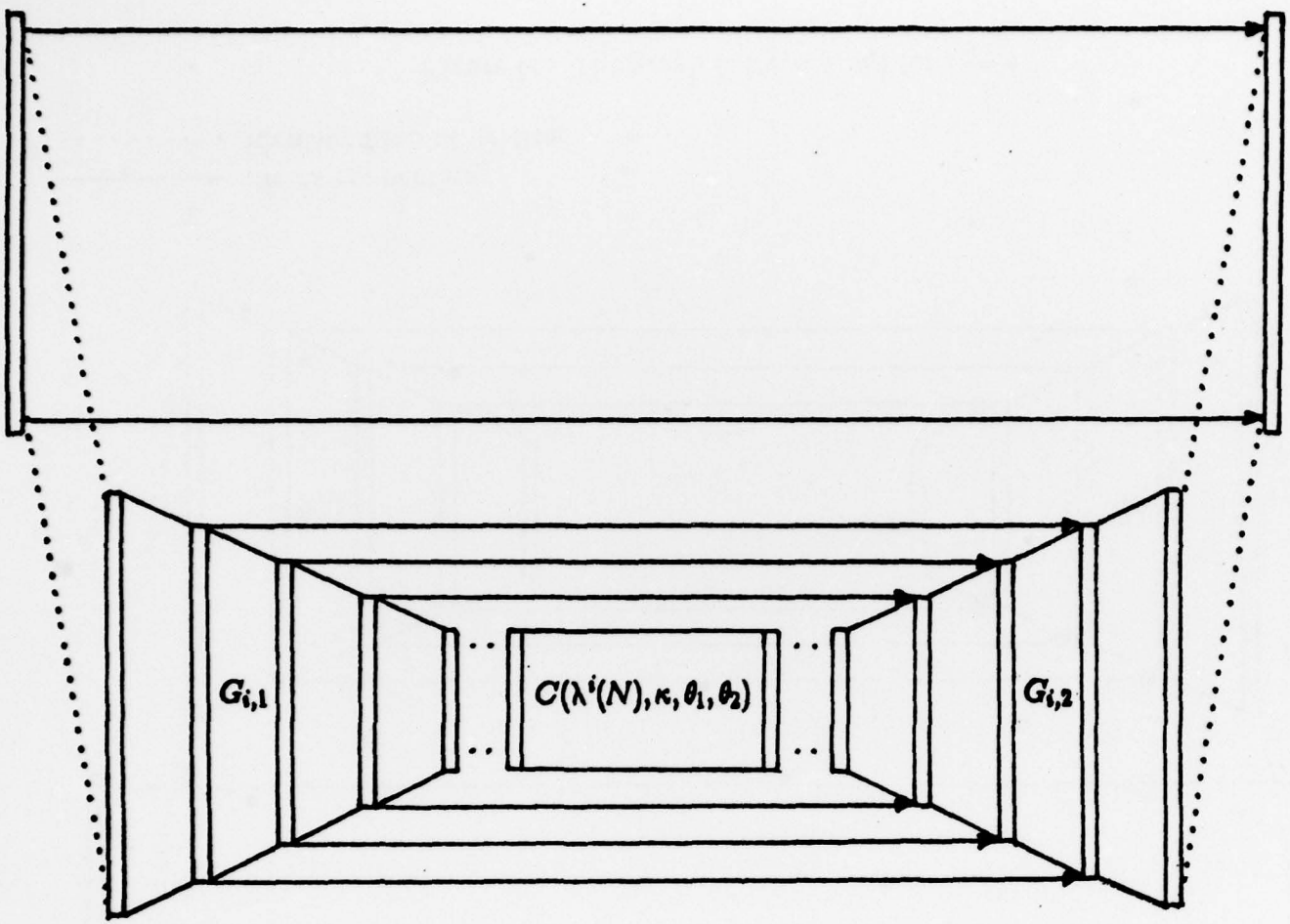
Figure 3: An N-superconcentrator with $O(N \cdot \log N)$ edges, $N = 16$
(all edges are directed towards the right)



direct connections

identification of vertices

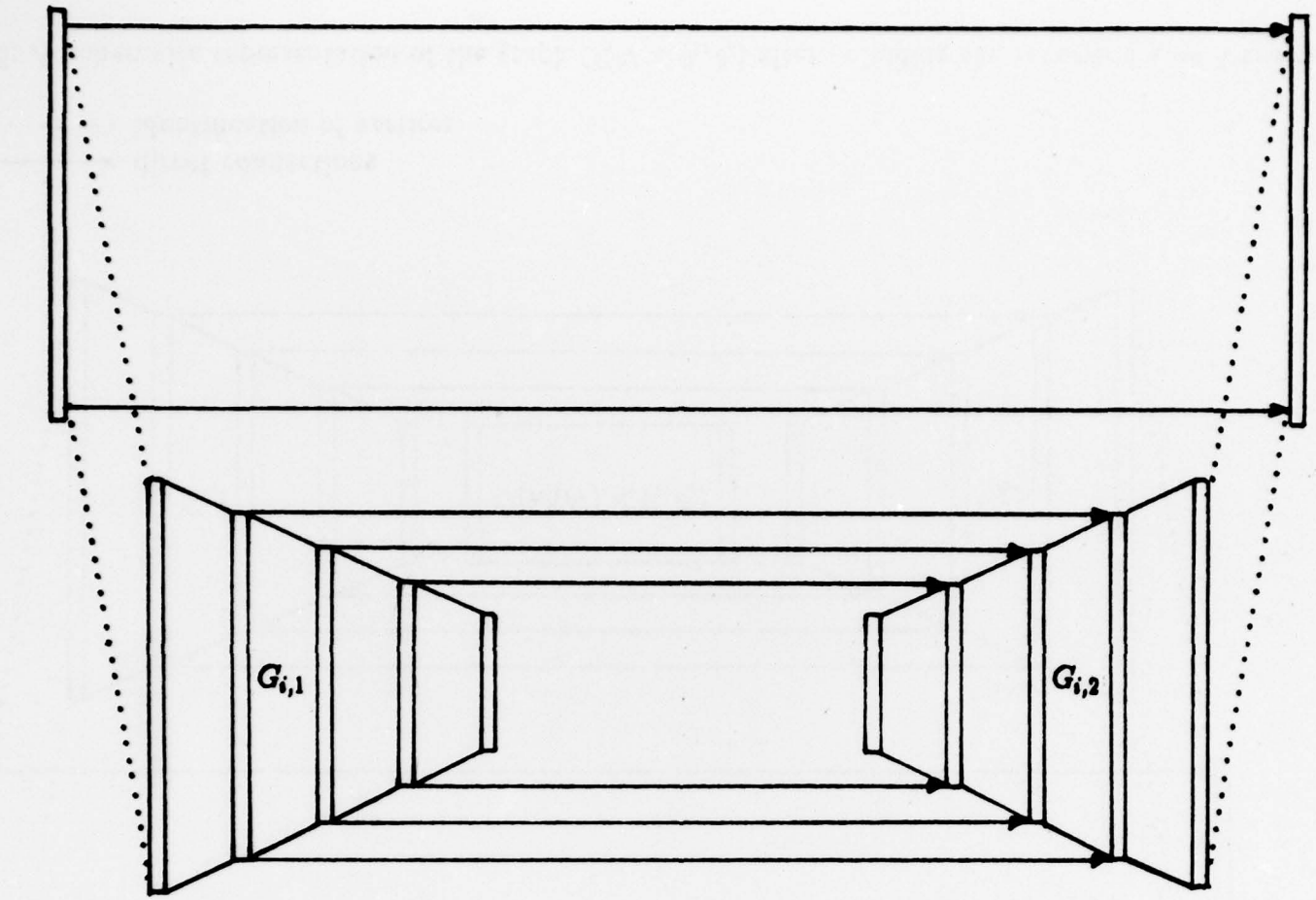
Figure 4: Pippenger's recursion scheme for the superconcentrator $C(N, \kappa, \theta_1, \theta_2)$



direct connections

identification of vertices

Figure 5: A schematic representation of the graph $C(N, \kappa, \theta_1, \theta_2)$ after unfolding the recursion $i = 4$ times



direct connections

identification of vertices

Figure 6: The graph $C_i(N, \kappa, \theta_1, \theta_2)$ for $i = 4$

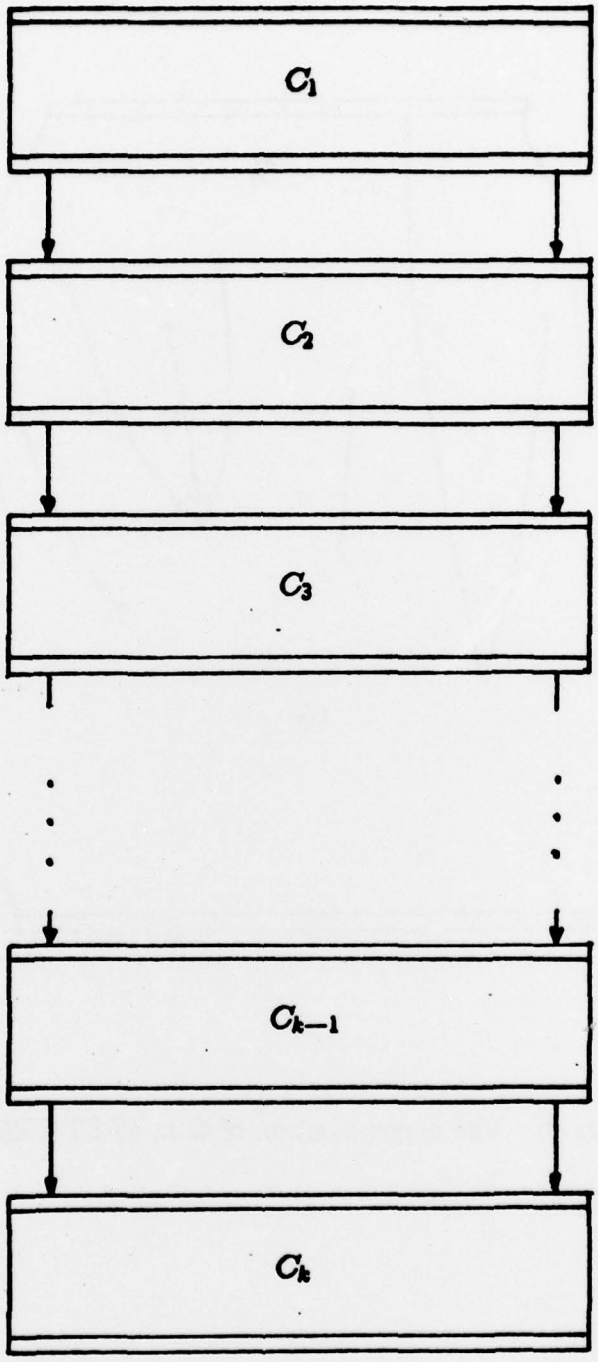


Figure 7: The graph $C(n, k)$

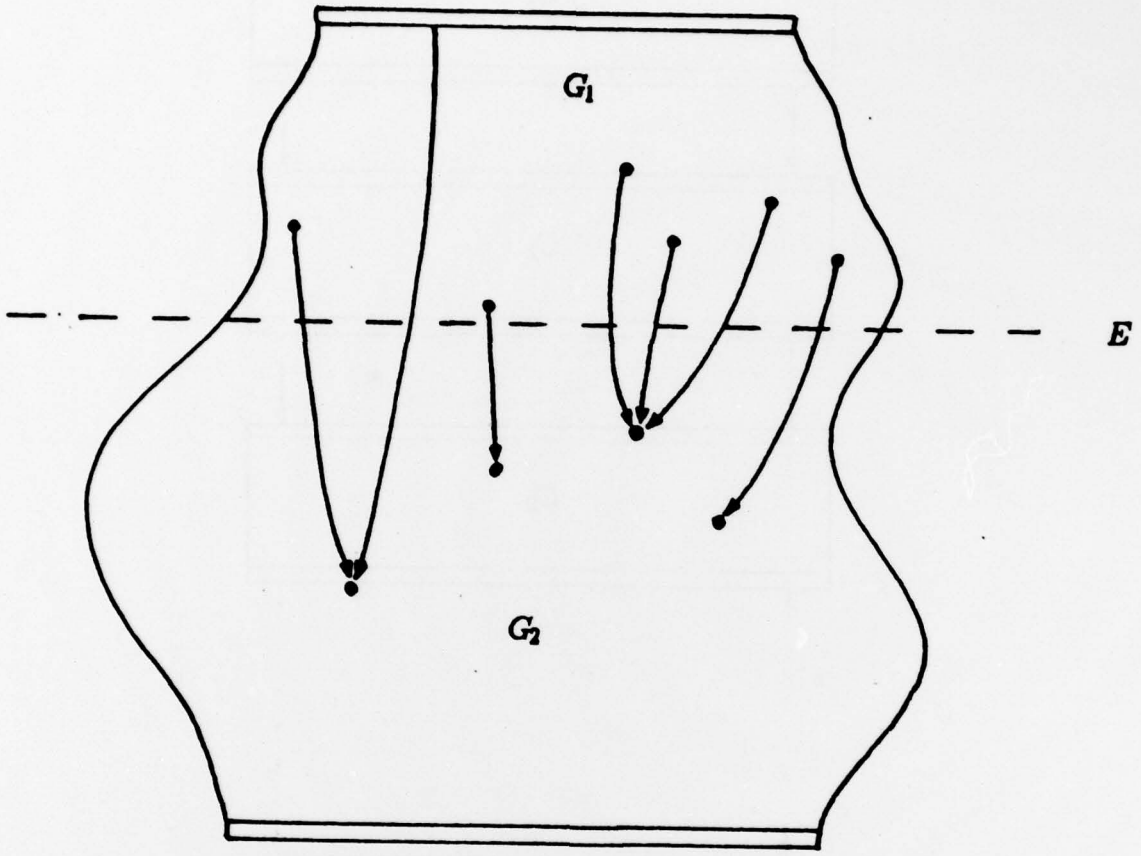


Figure 8: The decomposition of G in FAST-PEBBLE

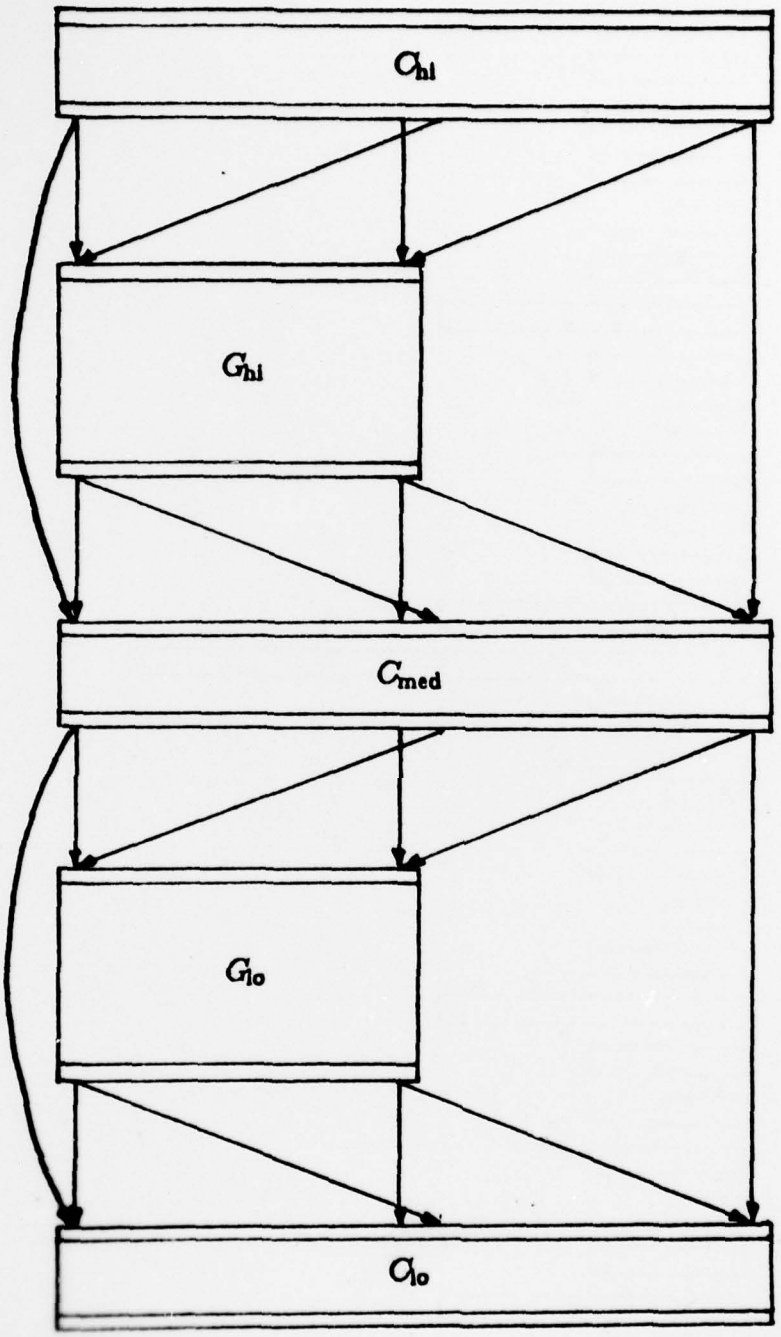


Figure 9: The recursion scheme for defining $G(n, k)$.

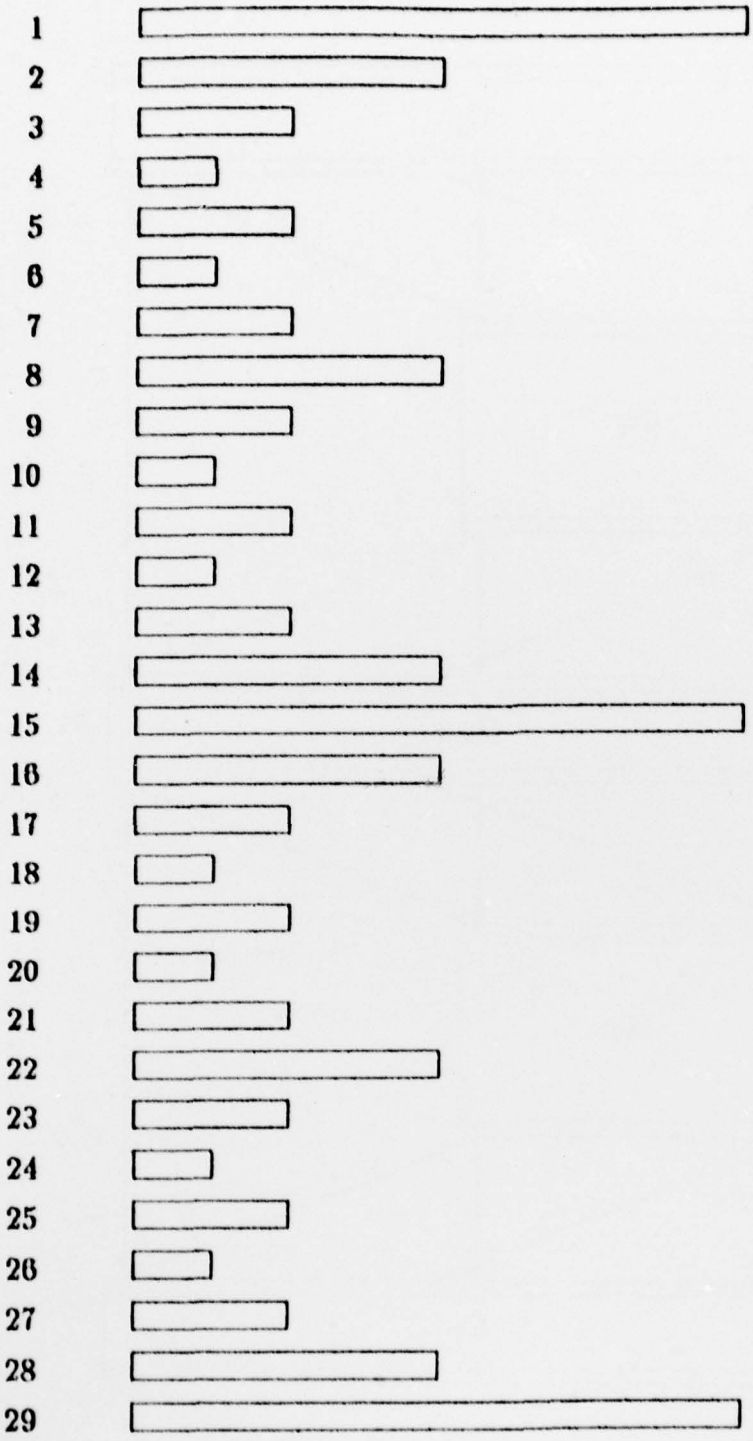


Figure 10: A schematic representation of $G(n,k)$ for $k = 4$ with level numbers